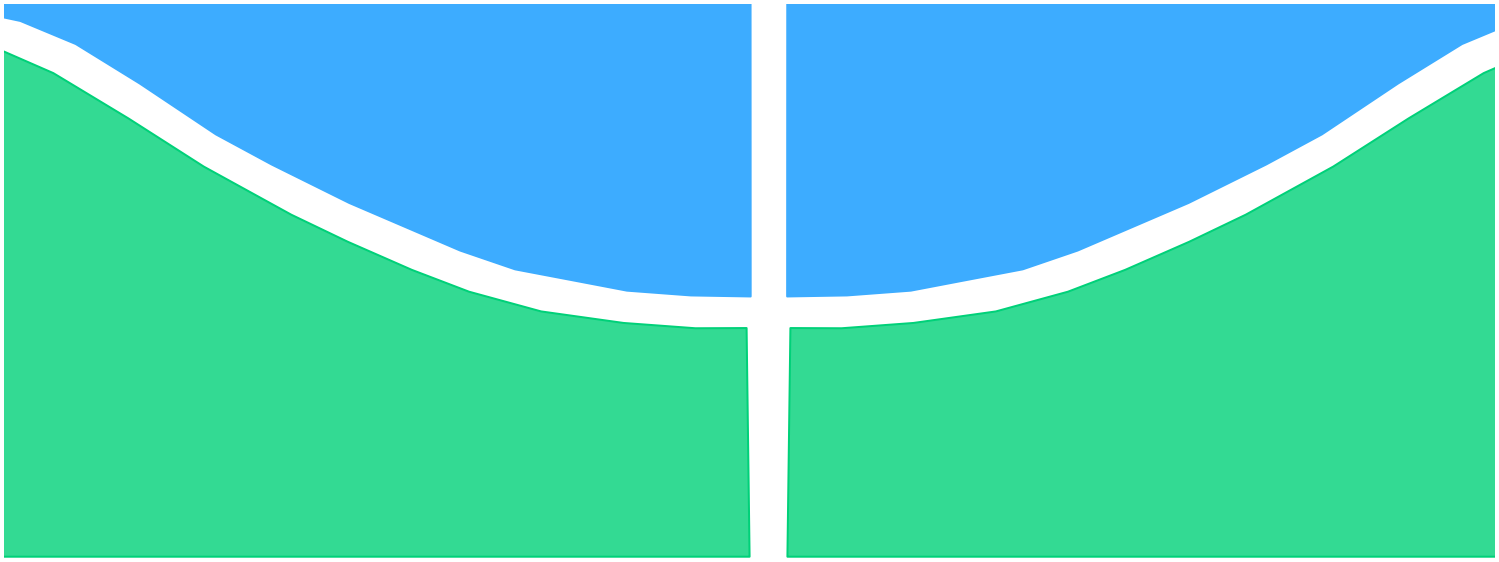


Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Curso de Engenharia de Software

PROPOSTA DE UM CONJUNTO DE COMPETÊNCIAS PARA UM TIME ÁGIL

Autor: Giancarllo Helion Frota Soares
Orientador: Msc. George Marsicano Corrêa

Brasília, DF
2014

Giancarllo Helion Frota Soares

PROPOSTA DE UM CONJUNTO DE COMPETÊNCIA PARA UM TIME ÁGIL

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Orientador: Msc. George Marsicano Corrêa

Co-Orientador: Dr. Wander Cleber Maria Pereira da Silva

Brasília, DF
2014

CIP – Catalogação Internacional da Publicação*

Soares, Giancarllo Helion Frota.
Proposta de um conjunto de competências para um time ágil / Giancarllo Helion Frota Soares. Brasília: UnB, 2014.
103 p. : il. ; 29,5 cm.

Monografia (Graduação) – Universidade de Brasília
Faculdade do Gama, Brasília, 2014. Orientação: Msc. George Marsicano Corrêa

1. Time ágil. 2. Competência. 3. Scrum I. Corrêa, George Marsicano. II. Msc.

CDU Classificação

- A ficha catalográfica oficial deverá ser solicitada à Biblioteca pelo aluno após a apresentação.

PROPOSTA DE UM CONJUNTO DE COMPETÊNCIAS PARA UM TIME ÁGIL

Giancarllo Helion Frota Soares

Monografia submetida como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software da Faculdade UnB Gama - FGA, da Universidade de Brasília, em (24/06/2014) apresentada e aprovada pela banca examinadora abaixo assinada:

Prof. Msc. George Marsicano Corrêa, UnB/ FGA
Orientador

Prof. Dr. Wander C. M. Pereira da Silva, UnB/ FGA
Co-Orientador

Profa. Dra. Rejane Maria da Costa Figueiredo, UnB/ FGA
Membro Convidado

Profa. Msc. Elaine Venson, UnB/ FGA
Membro Convidado

Brasília, DF
2014

AGRADECIMENTOS

Esse trabalho só pôde ser concluído graças à intervenção e ajuda de várias pessoas. Primeiramente agradeço a Deus por se mostrar sempre presente nos melhores e piores momentos que enfrentei, pela paciência, perseverança e discernimento que me concedeu e vem me concedendo e, principalmente, pelo amor e perdão incondicional que nos oferece.

Em segundo, agradeço a minha família que, independente das situações, sempre se mostrou a minha disposição para apoiar e ajudar no que fosse preciso e também pelo carinho, amor e acolhimento que tem comigo.

Devo agradecer também aos amigos tanto do curso quanto fora dele por serem responsáveis por tornar essa jornada menos difícil e mais divertida e pelo ajuda concedida em vários sentidos.

Agradeço aos meus professores pelos conhecimentos e experiências compartilhados que foram primordiais para minha formação acadêmica e profissional. Em especial, devo agradecer a meu orientador, Msc. George Marsicano Corrêa, e a meu co-orientador, professor Dr. Wander Cleber M. P. da Silva, pelos conselhos, direcionamentos e ajuda que culminaram no bom desenvolvimento deste trabalho e na solução dos problemas encontrados durante a sua confecção.

RESUMO

O desenvolvimento ágil de software apoiado pelo Manifesto Ágil surge como uma reação à engenharia de software tradicional. Como alicerce desse desenvolvimento, as metodologias ágeis suportam a aplicação e utilização dessa abordagem, seguindo os princípios e valores que ela prega. Essas metodologias atribuem novas características ao grupo de pessoas que irá desempenhar suas atividades através dela, sendo denominado por elas como equipe ágil. O time ágil, por fazer parte dessa equipe, herda essas qualidades e é quem mais as utiliza para realizar seu trabalho. Alguns livros, artigos e guias de métodos ágeis afirmam que o time ágil deve ter todas as competências necessárias para completar o trabalho sem depender de pessoas de fora desse time, mas nenhum deles mencionam quantas e quais são essas competências. Diante disso, este trabalho busca definir e validar um conjunto de competências iniciais para um time ágil. Para atingir esse propósito, foi realizado um levantamento bibliográfico por meio de uma revisão sistemática para identificar e conhecer as características de um time ágil e a escolha de um conceito de competência para ser utilizado. Um questionário foi elaborado para avaliar a importância das competências e produziu como resultados um *ranking* de competências baseado na importância dada pelos respondentes a cada umas delas, sendo constituído por 29 competências.

Palavras-chave: Time ágil. Competência. *Scrum.* XP

ABSTRACT

The agile software development supported by the Agile Manifesto arises as a reaction to traditional software engineering. As the basis of this development, agile methodologies support the application and use of this approach, following the principles and values that it preaches. These methodologies bring new characteristics to the group of people who will perform their activities through it, being termed as agile team. Some books, articles and guides of agile methods claim that agile team must have the competences necessary to complete the work without relying on people outside of this team, but none of them mention how many and what are these competences. Thus, this work seeks to define and validate a set of initial competences to an agile team. To achieve this purpose, we carried out a literature through a systematic review to identify and understand the characteristics of an agile team and choosing a concept of competence to be used. A questionnaire was designed to assess the importance of competences and as a result produced a ranking of competences based on the importance given by respondents to each one of them, consisting of 29 competences.

Keywords: Agile team. Competences. Scrum. XP

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE QUADROS

SUMÁRIO

1. INTRODUÇÃO

Este trabalho está inserido dentro do contexto de desenvolvimento ágil de software que será descrito e detalhado na Seção 1.1.

1.1. CONTEXTO

Com o surgimento do desenvolvimento ágil de software, a engenharia de software sofreu grandes modificações, o que em uma perspectiva tradicional observa-se o desenvolvimento de software através de um processo padronizado, controlável e previsível, esta visão é desafiada por uma perspectiva ágil que prioriza a singularidade, ambiguidade, complexidade, se contrapondo aos quesitos observados na anterior (DYBÅ, 2000). O objetivo dessa transição é promover flexibilidade e capacidade de resposta (NERUR e BALIJEPALLY, 2007).

Segundo Ambler (2002), para abordar os desafios encontrados pelos desenvolvedores de software, um grupo inicial de 17 metodólogos formaram a *Agile Software Development Alliance,* também conhecido como a *Agile Alliance,* em fevereiro de 2001. Uma particularidade interessante sobre este grupo é que todos os seus integrantes possuem uma origem de conhecimento diferente, mas foram capazes de chegar a um acordo que típicos metodólogos dificilmente chegariam ou acordariam (FOWLER, 2001). Este grupo definiu um manifesto chamado Manifesto Ágil para encorajar melhores caminhos para o desenvolvimento de software e, baseado neste manifesto, formularam princípios e valores que deram início e fundamentaram o desenvolvimento ágil de software e seus processos (AMBLER, 2002).

De acordo com Beck et al. (2001), por meio do Manifesto Ágil passa-se a valorizar:

- Indivíduos e interações mais que processos e ferramentas;
- Software em funcionamento mais que documentação abrangente;
- Colaboração com o cliente mais que negociação de contratos;
- Responder a mudanças mais que seguir um plano.

Esses são os valores que o manifesto prega para o desenvolvimento ágil de software, lembrando que, mesmo que haja valor nos itens mencionados à direita, o

grupo valoriza ainda mais os mencionados à esquerda (BECK et al, 2001). Geralmente, um manifesto está atrelado a um movimento político emergente que ataca aos conservadores e sugere uma mudança revolucionária e Pressman (1995) trata o desenvolvimento ágil exatamente assim, pois nele foi desenvolvido um esforço para sanar fraquezas reais e perceptíveis da engenharia de software convencional, mesmo não sendo indicada para todos os projetos.

Para Shore (2008), o desenvolvimento ágil se tornou popular e empresas como Google, Yahoo, Symantec, Microsoft, entre outros o utilizam, sendo uma lista crescente de instituições que estão adotando a este tipo de desenvolvimento, mesmo que nem sempre estejam o utilizando pelo motivo correto.

Brooks Jr (1995) em 1986 previu que até 1996 não haveria nenhuma solução, tecnologia ou técnica de gerenciamento que ofereça um grande aumento em produtividade, confiança ou simplicidade e foi o que aconteceu. Apesar de que estudos comprovem que equipes ágeis tem uma produtividade acima da média, o desenvolvimento ágil não é uma solução e não é recomendado apenas para obter uma melhor produtividade, pois sua equipe inicialmente deverá aprender como funciona e precisará de tempo para isso, podendo ficar mais lenta e menos produtiva (SHORE, 2008).

Sendo assim, é necessário perguntar antes se o desenvolvimento ágil de software vai ajudar a ter mais sucesso, pois ele não se trata de um processo específico, se trata de uma filosofia, uma maneira de pensar sobre o desenvolvimento de software descrita de uma forma canônica pelo Manifesto Ágil (SHORE, 2008).

Como alicerce desse tipo de desenvolvimento de software, as metodologias ágeis surgem para apoiar a utilização dessa abordagem. O *Scrum* e o XP (*eXtreme Programming*) são considerados os métodos ágeis mais adotados em projetos de desenvolvimento de software (FITZGERALD, et. al., 2006). Ambos trazem suas contribuições e particularidades, colaborando com a reação de mudança ao estilo tradicional.

O *Scrum* representa uma mudança radical para o planejamento e gerenciamento de projetos de software por dar autoridade de tomada de decisão a níveis de problemas operacionais e incertezas (MOE, et. al., 2010). Já o XP, apesar

de também abordar o planejamento e gerenciamento, enfatiza o uso de práticas mais voltadas para implementação como a programação em pares, integração contínua e uso de padrão de codificação.

Os métodos ágeis, herdando os valores do Manifesto Ágil, também tratam como prioridade o fator pessoa e trazem novas características ao grupo de pessoas que desempenha suas atividades através dessas metodologias. Essas novas características são a multifuncionalidade e a auto-organização, sendo difundida tanto no *Scrum* quanto no XP (SHORE, 2008) (SCHWABER e SUTHERLAND, 2013). A multifuncionalidade diz respeito à equipe ágil possuir todas as competências necessárias para completar o trabalho; já a auto-organização se trata da equipe ágil ser capaz de se gerenciar sem a necessidade de uma decisão gerencial ou da figura de um gestor.

Apesar dessa similaridade, o *Scrum* e o XP tratam a equipe ágil de maneira diferente. O *Scrum* define a equipe ágil ou a equipe *Scrum*, como prega a metodologia, em torno de três papéis: o *Product Owner*, o *Scrum Master* e o time de desenvolvimento. Já o XP define a equipe ágil em torno de quatro papéis comuns, podendo haver outros que são derivados desses principais, e são: clientes *On-Site*, programadores, testadores e *coaches*. Além dessa diferença, outra disparidade que pode ser notada através de estudo é que o *Scrum* delimita no time de desenvolvimento quem realmente é responsável por fazer o produto e o separa dos demais para evitar interferências desnecessárias; já o XP deixa essa decisão para a necessidade do projeto, podendo ter todos os seus papéis envolvidos em uma equipe (SHORE, 2008) (SCHWABER, 2004) (SHORE, 2008) (SCHWABER e SUTHERLAND, 2013).

Essa mesma distinção é defendida por Leffingwell (2011) que separa indivíduos envolvidos na implementação do produto dos detentores de conhecimento negocial e os que dão suporte a esses indivíduos.

Este trabalho destina-se a estudar o time de desenvolvimento, papel exclusivo do *Scrum*, utilizando as práticas do XP que foi denominado como time ágil.

1.2. PROBLEMA

O guia da metodologia *Scrum* (SCHWABER e SUTHERLAND, 2013) afirma que a equipe *Scrum* deve ser auto-organizável e deve possuir todas as competências necessárias para completar o trabalho sem depender de outros que não fazem parte da equipe. Não obstante, Cockburn e Highsmith (2001) afirmam que equipes ágeis de desenvolvimento focam em competências individuais como um fator crítico para o sucesso do projeto e que este foco está voltado para evoluir ou aumentar essas competências e os níveis de colaboração. Essas características estão extremamente ligadas ao time ágil, que além de fazer parte desta equipe, é quem mais utiliza desses atributos para realizar suas tarefas e construir o produto final.

Mesmo com guias e artigos, não são encontradas quais são as competências que o time ágil necessita. Alguns trabalhos e publicações também mencionam essa particularidade do time ágil possuir todas as competências que ele necessita e evoluí-las com progresso ou andamento do projeto, mas não é explícito quantas e quais são essas competências (MCHUGH, et. al., 2012) (MOE, et. al., 2010) (RUHNOW, 2007).

Diante disso, a pergunta de pesquisa deste trabalho é: Quais são as competências necessárias que um time ágil deve ter para desempenhar suas atividades?

A partir desse problema, este trabalho se propõe a responder essa questão.

1.3. OBJETIVOS

1.3.1. Objetivo geral

O objetivo principal deste trabalho foi propor um conjunto de competências para um time ágil, validado por meio de um instrumento de pesquisa.

1.3.2. Objetivos específicos

Para alcançar o objetivo principal foi necessário realizar os seguintes objetivos específicos:

- Contextualizar e caracterizar as metodologias ágeis;

- Caracterizar o método ágil *Scrum*;
- Caracterizar o método ágil XP;
- Detalhar o papel de um time de ágil;
- Identificar as características de um time ágil;
- Mapear os relacionamentos e atividades de um time ágil com as cerimônias do *Scrum*;
- Contextualizar e caracterizar as competências;
- Construir, aplicar um instrumento de pesquisa e analisar os dados coletados.

1.4. JUSTIFICATIVA

Diante dessa problemática, este trabalho se destina a propor um conjunto preliminar de competências para times ágeis, considerando esse time sendo o time de desenvolvimento do método *Scrum* utilizando as práticas do XP.

Além de ter um caráter inédito, já que não foi possível encontrar trabalhos com esse mesmo propósito, este trabalho visa contribuir tanto para o ambiente profissional quanto acadêmico. Para o lado acadêmico a contribuição está voltada para a criação de novas possibilidades de pesquisa nessa área e o desenvolvimento dessas competências encontradas no decorrer da graduação. Já para o profissional, é auxiliar na definição de um perfil para o time ágil que servirá como guia para a escolha de profissionais mais adequados com base nas suas competências.

Será utilizado como referencial para este trabalho as metodologias ágeis *Scrum* e XP, pois o *Scrum* é o único que realmente delimita em um time indivíduos que serão responsáveis pela implementação do produto, porém não menciona o que o time ágil faz para implementar esse produto, enquanto o XP traz por meio práticas técnicas maneiras de realizar isso.

1.5. METODOLOGIA DE PESQUISA

Este trabalho foi desenvolvido e orientado com base nos seguintes passos:

- Identificação do problema da pesquisa;
- Revisão sistemática;
- Elaboração da proposta inicial de competências;
- Elaboração do instrumento de pesquisa;
- Validação do instrumento de pesquisa;
- Ajustes do instrumento de pesquisa;
- Aplicação do instrumento de pesquisa;
- Coleta dos dados;
- Análise dos dados;
- Conclusões.

Os detalhes dessa metodologia utilizada e do desenvolvimento deste trabalho estão presentes na seção 3.

1.6. ORGANIZAÇÃO DO TRABALHO

Este trabalho está disposto e organizado em seções, sendo dividido em cinco seções. Nesta Seção 1 contempla-se introdução com uma contextualização e tema do trabalho, o problema, a justificativa, os objetivos e a metodologia de pesquisa.

Na Seção 2 é apresenta-se referencial teórico que traz as metodologias ágeis *Scrum* e XP e uma contextualização e conceitos de competências. Ao que se refere ao *Scrum*, são detalhados seus artefatos, cerimônias e papéis; já referente ao XP, são evidenciados seus valores, práticas técnicas e como ele aborda a arquitetura do sistema, ambos com foco no time ágil. No que tange às competências, são mostrados os conceitos e adesão de um neste trabalho.

Na Seção 3 apresenta-se a metodologia de pesquisa adotada neste trabalho, onde estão descritos a tipologia da pesquisa, a revisão sistemática, a elaboração, validação e alterações feitas no instrumento de pesquisa, assim como a sua disponibilização e determinação da sua consistência.

Na Seção 4 apresenta-se os resultados obtidos, constituída pela caracterização dos respondentes do questionário, os graus de importância obtidos

para as competências, análise desses graus e ordenação das competências por meio de um ranking de importância.

Na Seção 5 apresenta-se as considerações finais sobre este trabalho e as sugestão de trabalhos futuros.

2. REFERENCIAL TEÓRICO

Essa parte do trabalho destina-se ao referencial teórico, tendo como objetos as metodologias ágeis *Scrum* e XP e a caracterização de competências.

2.1. METODOLOGIAS ÁGEIS

As metodologias ágeis fazem parte da Engenharia de Software, que é uma área de conhecimento voltada para a especificação, desenvolvimento e manutenção de sistema de software. Pressman (1995) afirma que Engenharia de Software envolve quatro componentes básicos:

- Métodos – responsáveis por detalhar como é feita a construção de um software;
- Estimativas – focadas na parte de planejamento, contemplam análises de requisitos, arquitetura, entre outros;
- Ferramentas – sustentam a aplicação dos métodos;
- Procedimentos – sequência em que os métodos são aplicados, relacionando cada método à ferramenta que o suporta.

Já Shore (2008) define metodologias ou métodos ágeis como processos que suportam a filosofia ágil, sendo constituídos de elementos individuais chamados práticas. Métodos ágeis combinam-se de várias maneiras, destacando as partes que suportam a filosofia ágil, descartando o que não interessa ou o restante e misturando novas ideias, o que resulta em um pacote enxuto, poderoso e que se auto-reforça continuamente. Essa definição também colabora com a de que metodologias ágeis são um conjunto de práticas que seguem os princípios do Manifesto Ágil (BECK et al., 2001).

Buscando entender o propósito desses métodos, apresentam-se esses princípios, que segundo Beck et al. (2001) são:

- A maior prioridade é satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado;

- Mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagem das mudanças visando vantagem competitiva para o cliente;
- Entregar frequentemente software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo;
- Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto;
- Construir projetos em torno de indivíduos motivados. Dê a eles o ambiente e o suporte necessário e confie neles para fazer o trabalho;
- O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através de conversa face a face;
- Software funcionando é a medida primária de progresso;
- Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente;
- Contínua atenção à excelência técnica e bom design aumenta a agilidade;
- Simplicidade – a arte de maximizar a quantidade de trabalho não realizado – é essencial;
- As melhores arquiteturas, requisitos e *designs* emergem de equipes auto-organizáveis;
- Em intervalos regulares, a equipe reflete sobre como se torna mais eficaz e então refina e ajusta seu comportamento de acordo.

Segundo Highsmith e Cockburn (2001), a maioria das metodologias ágeis não possuem nada de novo e o que as diferencia das tradicionais são o enfoque e os valores difundidos. Algumas metodologias ágeis vieram até antes do Manifesto Ágil, como *Extreme Programming* que data dos anos 80, tendo o manifesto o papel de proporcionar a troca de ideias que acabou gerando os princípios e valores para esta abordagem.

Dentre as várias metodologias ágeis disponíveis, este trabalho se restringirá à metodologias *Scrum* e XP.

2.1.1. Scrum

O termo *Scrum* surgiu de uma atividade que ocorre durante uma partida de *rugby*, onde um grupo de jogadores faz uma formação em torno da bola e seus companheiros de equipe trabalham juntos para avançar com a bola em direção ao fundo do campo adversário (PRESSMAN, 1995). A utilização desse termo se deu pela primeira vez por Nonaka e Takeuchi, que no capítulo "*Moving the Scrum Downfield"* descrevem seis características que, se usadas conjuntamente, produzem um conjunto dinâmico e são utilizadas pelas empresas de ponta no gerenciamento de seus projetos (TAKEUCHI e NONAKA, 1986).

Segundo Alliance (2013), *Scrum* é um framework ágil para realização de projetos complexos e originalmente foi formalizado para projetos de desenvolvimento de software, porém seu funcionamento pode ser bem aplicado em qualquer escopo complexo ou contexto inovador de trabalho. Suas possibilidades são infinitas e sua aplicação é definitivamente simples.

O *Scrum* é o mais conhecido dos métodos ágeis pela sua ênfase dada ao gerenciamento de projeto. Ele reúne atividades de monitoramento e *feedback*, em geral, reuniões rápidas e diárias com toda a equipe para identificar e corrigir quaisquer problemas e/ou impedimentos no processo de desenvolvimento (SCHWABER, 2004).

O *Scrum* engloba todas as suas práticas em torno de um esqueleto de um processo iterativo e incremental (SCHWABER, 2004). Esse esqueleto está disposto na Figura 1.

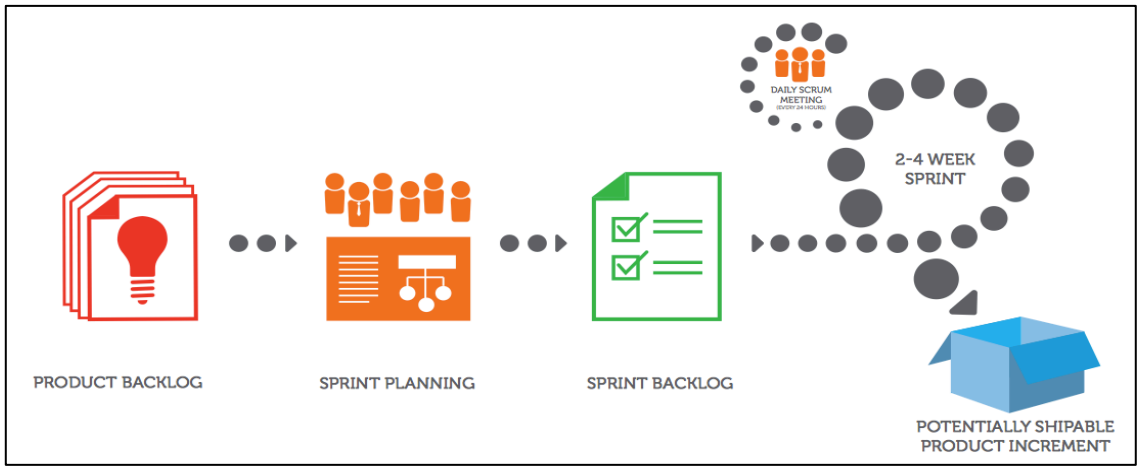


Figura 1. Esqueleto do *Scrum*. Fonte: (ALLIANCE, 2013).

Na Figura 1 é possível identificar dois círculos, um pequeno e um maior. O maior representa uma iteração de desenvolvimento de atividades que ocorrem uma após a outra chamada *Sprint*, que geralmente pode durar duas, quatro semanas até trinta dias consecutivos. O círculo menor corresponde às inspeções diárias que ocorrem durante uma *Sprint* chamada *Daily Scrum Meeting*, onde os membros da equipe se reúnem para inspecionar cada uma das atividades e fazer as devidas adaptações (SCHWABER, 2004).

De uma forma bem simples e objetiva, a Figura 1 permite visualizar o ciclo de um projeto *Scrum* que se repete várias vezes até que o último incremento do produto final seja concluído, considerando elementos importantes para este método ágil como, por exemplo, o *Product Backlog e o Sprint Backlog*, que são artefatos exclusivos do *Scrum*, e o *Sprint Planning e o Daily Scrum Meeting*, que são cerimônias ou regras exclusivas do *Scrum*. Vale ressaltar também que esses artefatos e regras envolvem papéis específicos e exclusivos desse método.

2.1.2. Equipe Scrum

A equipe *Scrum* é composta por três papéis: *Product Owner*, *Scrum Master* e o time de desenvolvimento. As responsabilidades de gerenciamento no projeto são divididas entre esses três papéis (SCHWABER, 2004).

As pessoas que seguem e preenchem esse papéis são aqueles que se comprometem com o projeto, podem haver outros que estejam interessados no projeto, mas eles não se encaixam na equipe. O *Scrum* faz essa distinção para garantir que aqueles responsáveis pelo projeto tenham autoridade para fazer o que é necessário para o sucesso dele e que não seja dada essa liberdade para aqueles que não tem o mesmo nível de comprometimento para que não interfiram desnecessariamente (SCHWABER, 2004).

Nas próximas seções serão descritos os papéis que compõem a equipe *Scrum*.

2.1.2.1. Time de desenvolvimento

O time de desenvolvimento é um grupo de profissionais que realizam o trabalho de entregar uma versão funcional e usável que potencialmente incrementa o produto “pronto” ao final de cada *Sprint*, sendo apenas seus integrantes responsáveis por construir esse incremento (SCHWABER e SUTHERLAND, 2013).

Times de desenvolvimento são auto-organizáveis e multifuncionais, pois escolhem a melhor forma para completarem seu trabalho sem necessitar de uma direção gerencial ou de fora do time e possuem todas as competências necessárias para completar o trabalho sem depender de novas contratações (ALLIANCE, 2013) (SCHWABER e SUTHERLAND, 2013). O *Scrum* também requer que o time de desenvolvimento tenha essas características, que tenha todas as habilidades e competências necessárias para entregar cada incremento do produto, que seus membros estejam disponíveis para o projeto em tempo integral, não dividindo seu tempo em outros projetos, e que se auto organizem para realizar o objetivo da *Sprint*, produzindo um novo incremento do produto de acordo com o plano da *Sprint* (ALLIANCE, 2013).

O time de desenvolvimento é estruturado e autorizado pela organização para organizar e gerenciar o seu próprio trabalho, sendo resultante dessa sinergia a eficiência e a eficácia do time como um todo (SCHWABER e SUTHERLAND, 2013). Os integrantes individualmente podem ter habilidades especializadas e/ou áreas de especialização, mas a responsabilidade e o comprometimento pertence ao time como um todo.

Para este trabalho, o time de desenvolvimento será tratado como time ágil, utilizando também as práticas do XP.

2.1.2.2. *Product Owner*

O *Product Owner*, ou dono do produto por tradução literal, representa o cliente ou aquele que detém o conhecimento a nível de negócio do projeto, porém se comporta de uma forma diferente daquele cliente de projetos tradicionais. De acordo com os princípios do Manifesto Ágil, o *Product Owner* está idealmente localizado junto ao time ágil e participa diariamente com esse time e nas atividades dele (LEFFINGWELL, 2011).

Segundo Cohn (2009), o *Product Owner* é aquela pessoa que garante que o time ágil está voltado para o objetivo certo, sendo um papel determinante para o sucesso do projeto. Ele é o único responsável por determinar, priorizar e manter os requisitos do usuário, ou seja, gerenciar o *Product Backlog*. Segundo Schwaber e Sutherland (2013), o gerenciamento do *Product Backlog* se baseia em:

- Expressar claramente os itens do *Product Backlog*;
- Ordenar os itens do *Product Backlog* para alcançar melhor as metas e missões;
- Garantir o valor do trabalho realizado pelo time ágil;
- Garantir que o *Product Backlog* seja visível, transparente, claro para todos, e mostrar o que a equipe Scrum vai trabalhar a seguir;
- Garantir que o time ágil entenda os itens do *Product Backlog* no nível necessário.

O *Product Owner* pode representar vários *stakeholders*, mas só pode ser uma pessoa, podendo expressar o desejo deles, contudo, caso esse desejo seja de alteração no *Product Backlog*, eles devem convencê-lo disso (ALLIANCE, 2013) (SCHWABER e SUTHERLAND, 2013).

De forma resumida, o foco do *Product Owner* está no retorno do investimento, sendo responsável por garantir que o projeto tenha um bom retorno sobre o investimento feito (SCHWABER, 2004) (COHN, 2009).

2.1.2.3. *Scrum Master*

O *Scrum Master* representa um “servo-líder” que ajuda o resto da equipe *Scrum* a seguir o processo, garantindo que eles aderem à teoria, práticas e regras do *Scrum* (SCHWABER e SUTHERLAND, 2013).

O *Scrum Master* não pode dizer “você está demitido”, e sim “decidi que nós tentaremos usar *Sprints* de duas semanas para o próximo mês”, o que faz dele a pessoa que ajuda o time ágil no uso do *Scrum*. Pode ser comparável a um treinador que ajuda o time com um regime de exercícios para que eles realizem todos de forma correta, provê motivação ao time certificando que ninguém burle ou ignore

tarefas difíceis, mas não pode realizar uma tarefa que alguém não fez, pois possui autoridade limitada, sendo determinada pelo time ágil (COHN, 2009).

Ele também ajuda aqueles que estão fora da equipe *Scrum* a entender quais as suas interações com a equipe são úteis e quais não são para que não haja interferências desnecessárias na rotina de trabalho e maximizar o valor criado (SCHWABER e SUTHERLAND, 2013).

Em resumo, o *Scrum Master* pode contribuir com todos, com o *Product Owner,* com o time ágil e com a organização, atuando e servindo de várias maneiras para que seja garantido o entendimento e bom funcionamento do *Scrum* (SCHWABER e SUTHERLAND, 2013).

2.1.3. Artefatos do *Scrum*

O *Scrum* define e utiliza três artefatos que são usados constantemente durante seu ciclo e serão descritos nas seguintes seções.

2.1.3.1. *Product Backlog*

Backlog, por tradução literal, significa reserva ou acúmulo, o que pode direcionar o entendimento de *Product Backlog.*

Uma equipe *Scrum* renuncia à lentidão e prematuridade de uma fase de requisitos em favor de uma abordagem imediata, onde descrições de funções de alto nível podem ser obtidas mais cedo, mas, no início, são minimamente descritas e progressivamente refinadas com o andamento do projeto (COHN, 2009). Essas descrições são documentadas em um *Product Backlog*, que é uma lista de requisitos ou funcionalidades desejadas para o sistema ou produto que está sendo desenvolvido (COHN, 2009) (SCHWABER, 2004).

O *Product Backlog* nunca é completo, pois ele é usado inicialmente no plano de projeto para gerar estimativas iniciais dos requisitos, que serão evoluídos ou modificados de acordo com a evolução do produto ou do ambiente que será utilizado (SCHWABER, 2004). Com isso, percebe-se o quanto ele é dinâmico; o gerenciamento muda constantemente para identificar o que o produto precisa ter para ser apropriado, competitivo e útil; itens são adicionados, removidos e/ou

repriorizados a cada *Sprint* de acordo com o que é aprendido sobre o produto, o usuário, o time e assim por diante (SCHWABER, 2004)(COHN, 2009).

2.1.3.2. *Sprint Backlog*

Assim como o *Product Backlog*, o *Sprint Backlog* também é lista de requisitos de um produto, mas que é destinado para uma *Sprint*. O *Sprint Backlog* define o trabalho do time ágil em uma *Sprint*, sendo composto por tarefas que o time definiu a partir do que estava determinado no *Product Backlog* para aquela *Sprint* visando transformar aqueles itens a fazer em um incremento de funcionalidade do produto (SCHWABER, 2004).

O *Sprint Backlog* é alterado apenas pelo time ágil e deve ser uma imagem de tempo real bem visível do trabalho que o time planejou realizar durante a *Sprint* (SCHWABER, 2004).

2.1.3.3. Incremento de funcionalidade do produto

O *Scrum* precisa de times ágeis que construam um incremento de funcionalidade do produto a cada *Sprint*, que deve ser potencialmente adequado para entrega, pois o *Product Owner* pode escolher por implantar imediatamente essa parte realizada, o que requer que este incremento seja rigorosamente testado, bem estruturado e de código bem escrito (SCHWABER, 2004).

Quando o incremento se encontra nessas condições, é posto como executável e permite que a funcionalidade do usuário seja documentada, contribuindo para o manual do produto (SCHWABER, 2004).

O incremento de funcionalidade do produto é de responsabilidade do time ágil, podendo ser uma funcionalidade ou parte dela, e é gerado ao término de uma *Sprint* com realização das tarefas presentes no *Sprint Backlog*.

2.1.4. Cerimônias do *Scrum*

Todos os eventos ou cerimônias que são usados no *Scrum* criam uma rotina e minimizam a necessidade de se criar reuniões não definidas pela metodologia

(SCHWABER e SUTHERLAND, 2013). Essas cerimônias são eventos *time-boxed*, ou seja, possuem um tempo de duração máxima. A duração específica para cada evento não pode ser reduzida ou aumentada e o evento pode terminar sempre que o seu propósito é alcançado, o que garante que uma quantidade adequada de tempo seja gasta sem permitir perdas no processo (SCHWABER e SUTHERLAND, 2013).

Mudanças nessas cerimônias só podem ser originadas pelo time ágil e intermediadas pelo *Scrum Master*, que deve estar ciente de que o time e todos os envolvidos entendam como o *Scrum* funciona o suficiente para que sejam hábeis e conscientes ao realizar as alterações (SCHWABER, 2004).

O *Scrum* prega e utiliza cinco cerimônias que serão descritas nas seguintes seções.

2.1.4.1. *Sprint Planning Meeting*

O trabalho que será realizado na *Sprint* é planejado no *Sprint Planning Meeting*. Nesta reunião a equipe *Scrum* colabora para selecionar e entender o trabalho que será feito na próxima *Sprint (*ALLIANCE*,* 2013).

Este evento possui um prazo fixo ou duração máxima de oito horas e consiste em dois momentos de quatro horas cada. O primeiro momento é para seleção do *Product Backlog* e o segundo para preparar o *Sprint Backlog*.

Segundo Ken Schwaber (2004), o *Sprint Planning Meeting* deve seguir os seguintes critérios:

- Os participantes dessa reunião são o *Scrum Master*, o *Product Owner* e o time ágil, onde convidados adicionais podem ser chamados apenas por essas pessoas para providenciar informações de domínio de negócio, tecnológico ou conselhos, devendo ser descartados após a informação ser transmitida;
- O *Product Owner* precisa preparar um prévio *Product Backlog* para a reunião. Em caso de ausência do *Product Owner*, é tarefa do *Scrum Master* fazer um prévio *Product Backlog* e substituir o *Product Owner*;

- A finalidade do primeiro momento ou as primeiras quatro horas da reunião é para time ágil selecionar aqueles itens do *Product Backlog* que ele acredita poder se comprometer a transformá-los em um incremento de funcionalidade do produto;
- O time ágil pode fazer sugestões, mas a decisão de que parte do *Product Backlog* pode constituir a *Sprint* é responsabilidade do *Product Owner*;
- É responsabilidade do time ágil determinar quanto do *Product Backlog* que o *Product Owner* deseja ele tentará fazer durante a *Sprint;*
- No primeiro momento, será disponível quatro horas para análise do *Product Backlog*. Uma análise posterior deve ser feita durante a *Sprint*, pois um *Product Backlog* com estimativas imprecisas pode não ser completamente entendido durante essa primeira parte da reunião o que pode resultar na incapacidade do time ágil completar os itens selecionados do *Product Backlog*;
- O segundo momento do *Sprint Planning Meeting* ocorre imediatamente após o primeiro e também possui um *time-boxe* de quatro horas;
- O *Product Owner* precisa estar disponível para o time ágil durante este segundo momento para responder questões que o time pode ter sobre o *Product Backlog*;
- Cabe ao time ágil, atuando sozinho e sem qualquer direção de fora, descobrir durante o segundo momento como ele vai transformar o *Product Backlog* selecionado em um incremento de funcionalidade do produto. Ninguém é autorizado de fazer qualquer coisa, a não ser observar e responder questões que o time ágil fizer;
- A saída do segundo momento do *Sprint Planning Meeting* é uma lista de tarefas, estimativas de tarefas e atribuições, chamada de *Sprint Backlog*, que farão o time ágil iniciar o trabalho de desenvolvimento da funcionalidade. A lista de tarefas não precisa estar completa, mas precisa ser completa o suficiente para refletir o compromisso mútuo por parte de todos participantes e para suportá-los durante a primeira parte da *Sprint*, enquanto o time providencia mais tarefas para o *Sprint Backlog*.

Após todos esses passos e detalhes que o *Sprint Planning Meeting* deve conter, ele esclarece tópicos como o objetivo da *Sprint*, o que pode ser entregue como resultado da próxima *Sprint* e como será o trabalho necessário para entregar o incremento será realizado (SCHWABER e SUTHERLAND, 2013).

2.1.4.2. *Daily Scrum Meeting*

De acordo com o Schwaber e Sutherland (2013), o *Daily Scrum Meeting* é uma reunião diária que o time ágil faz para inspecionar o progresso em direção ao objetivo da *Sprint* e completar o trabalho no *Product Backlog*. Essa reunião aumenta a probabilidade do time ágil atingir o objetivo da *Sprint* e reforçar o entendimento sobre como pretende trabalhar em conjunto para criar um incremento esperado até o final da *Sprint*.

O *Daily Scrum Meeting* possui duração máxima de 15 minutos independentemente da quantidade de membros do time ágil. Segundo Ken Schwaber (2004), deve seguir os seguintes critérios:

- Manter o *Daily Scrum* no mesmo lugar e no mesmo horário todos os dias de trabalho. O *Daily Scrum* é a melhor coisa a ser realizada primeiro no dia, pois a primeira coisa que os membros do time ágil fazem ao chegar no trabalho é pensar sobre o que eles fizeram no dia anterior e o que vão fazer hoje;
- Todos os membros do time ágil devem comparecer. Caso algum membro não possa comparecer, o membro ausente deve participar da reunião por telefone ou outro membro do time ágil deve reportar o progresso do ausente;
- Os membros do time ágil devem estar prontos. O *Scrum Master* inicia a reunião na hora determinada independente de quem está presente. Todos o membros que se atrasarem pagam um dólar ao *Scrum Master* imediatamente;
- O *Scrum Master* dá início à reunião começando pela pessoa que está a sua esquerda e prosseguindo em sentido anti-horário ao redor da sala até que todos tenham relatado;

- Cada membro do time ágil deve responder apenas três questões: o que ele fez desde o último *Daily Scrum*, o que ele irá fazer agora até o próximo *Daily Scrum* e o que o impede de realizar seu trabalho da melhor maneira possível;
- Durante o *Daily Scrum*, apenas uma pessoa fala de cada vez. Esta pessoa é a que relata o seu próprio progresso, onde o restante escuta e não há conversas paralelas;
- Quando os membros do time ágil relatam algo que interessa a outro membro ou precisa da assistência de outro membro, qualquer membro do time pode imediatamente providenciar que as partes interessadas se reúnam após o *Daily Scrum* para marcar uma reunião.

Essas reuniões diárias melhoram a comunicação, identificam e removem impedimentos para o desenvolvimento, destacam e promovem rápidas tomadas de decisão e melhoram o nível de conhecimento do time ágil, sendo uma reunião chave para inspeção e adaptação do trabalho realizado e do que será feito (SCHWABER e SUTHERLAND, 2013).

2.1.4.3. *Sprint*

Como todo processo ágil, o *Scrum* é uma abordagem iterativa e incremental de desenvolvimento de software, possuindo a característica de implementar parte por parte ou função por função do produto e o melhorando constantemente (COHN, 2009). Para exercer essas características, o *Scrum* utiliza a *Sprint* para realizar o trabalho que deve ser feito.

O termo *Sprint* faz menção a uma arrancada ou uma corrida, que é como o time ágil se comporta para realizar as tarefas que precisam ser feitas. O coração do *Scrum* é a *Sprint*, que é um *time-boxe* de um mês ou menos, aonde uma versão incremental potencialmente utilizável do produto é criada (SCHWABER e SUTHERLAND, 2013). Uma nova *Sprint* se inicia após a conclusão da anterior.

Segundo Ken Schwaber (2004), uma *Sprint* deve seguir os seguintes critérios:

- O time ágil pode procurar conselhos de fora, ajuda, informação e apoio durante a *Sprint;*

- Ninguém pode providenciar conselhos, instruções, comentários, ou direção ao time ágil durante a *Sprint*, pois o time é completamente auto-gerenciável;
- O time ágil se compromete com o *Product Backlog* durante o *Sprint Planning Meeting*, por isso ninguém pode alterar este *Product Backlog* até o final da *Sprint*, devendo permanecer congelado;
- Se a *Sprint* não provar ser viável, o *Scrum Master* pode em caráter anormal terminá-la e iniciar um novo *Sprint Planning Meeting* para definir a próxima *Sprint*. O *Scrum Master* pode fazer essa mudança de acordo com sua própria vontade ou por pedido do time ágil ou do *Product Owner*. A *Sprint* pode provar ser não viável se a tecnologia se tornar inexequível, se as mudanças nas condições do negócio naquela *Sprint* não terem valor ao negócio, ou se o time ágil for interferido por alguém de fora dele durante a *Sprint*;
- Se o time ágil se sentir incapaz de completar tudo o que foi prometido no *Product Backlog* para aquela *Sprint*, o *Product Owner* poderá ser consultado para remover aqueles itens da *Sprint* atual. Se muitos itens precisarem ser removidos, a *Sprint* perderá seu valor e sentido, podendo o *Scrum Master* em caráter anormal terminá-la como previsto anteriormente;
- Se o time ágil determinar que pode se comprometer mais com o *Product Backlog* durante a *Sprint* do que foi selecionado no *Sprint Planning Meeting*, o *Product Owner* pode ser consultado para que itens do *Product Backlog* sejam adicionados à *Sprint*;
- O time ágil tem duas responsabilidades administrativas durante a *Sprint*: ele deve participar do *Daily Scrum Meeting* e deve manter o *Sprint Backlog* atualizado e disponível em uma pasta pública em um servidor público, visível a todos.

O *Scrum* não diz que o time ágil codifica ou testa, ele apenas requer que o time entregue um código de alta qualidade e de potencial para ser entregue a cada final de *Sprint*, deixando isso implícito ou a cargo do próprio time ágil decidir o que deve fazer (COHN, 2009).

Diante disso, Leffingwell (2011) afirma que o time ágil escreve o código, podendo o desenvolvedor trabalhar no modelo de programação em pares com outro

desenvolvedor ou testador, e o testa, sendo o teste presente desde quando o código é previsto até quando ele é liberado. Em todo o caso, Leffingwell (2011) diz que a responsabilidade do time ágil é a mesma, e isso inclui as seguintes atividades:

- Escrever o código;
- Escrever os testes enquanto o código está sendo escrito;
- Escrever métodos necessários para suportar os testes que foram ou serão escritos;
- Testar o código contra os testes escritos;
- Realizar *check-in* de código novo no repositório compartilhado;
- Realizar *check-in* de casos de teste no repositório compartilhado;
- Integrar os testes em ambiente de teste contínuo.

Cada *Sprint* pode ser considerada como um projeto que não deve estender mais que um mês, pois, tendo a definição do que é para ser feito, um plano projetado e flexível para guiar a construção, o trabalho e o resultado do produto, quando a *Sprint* é muito longa, a definição pode mudar, a complexidade pode aumentar e o risco crescer (SCHWABER e SUTHERLAND, 2013). *Sprints* permitem previsibilidade que ajuda a inspeção e adaptação do progresso em direção às metas determinadas.

2.1.4.4. *Sprint Retrospective Meeting*

O *Sprint Retrospective Meeting* é o momento que a equipe *Scrum* tem para se inspecionar e criar um plano de melhorias que serão aplicadas na próxima *Sprint* (SCHWABER e SUTHERLAND, 2013). É uma reunião com duração máxima de três horas, podendo ser menor no caso de *Sprints* menores. Este evento deve ocorrer antes do *Sprint Planning Meeting* e depois do *Sprint Review*.

Segundo Ken Schwaber (2004), o *Sprint Retrospective Meeting* deve seguir os seguintes critérios:

- Devem participar apenas o time ágil e o *Scrum Master*, sendo a presença do *Product Owner* opcional;

- Ao iniciar a reunião, os membros do time ágil respondem duas perguntas: o que correu bem durante a última *Sprint* e o que pode ser melhorado para a próxima;
- O *Scrum Master* escreve de forma resumida as respostas do time ágil;
- O time ágil prioriza a ordem que ele deseja falar sobre as potenciais melhorias;
- O *Scrum Master* não participa desta reunião para responder, mas pode facilitar a busca do time ágil por melhores maneiras para o processo do *Scrum* funcionar.

Ao término do *Sprint Retrospective Meeting,* a equipe *Scrum* deve possuir as melhorias que serão implementadas na próxima *Sprint*, pois implementá-las é uma maneira de adaptação à inspeção que a equipe *Scrum* faz a si própria (SCHWABER e SUTHERLAND, 2013).

2.1.4.5. Sprint Review Meeting

No final de cada *Sprint*, a equipe *Scrum* e os *stakeholders* se reúnem e revisam o que foi produzido, tendo como ponto central de discussão o incremento do produto feito (ALLIANCE*,* 2013). Essa reunião é denominada *Sprint Review Meeting* e tem duração máxima de quatro horas de duração, podendo ser menor no caso de *Sprints* menores.

Normalmente, é sábio e útil para os *stakeholders* comparecerem a esta reunião, pois são interessados diretamente no produto e por se tratar de uma reunião informal que transparece aonde o time ágil se encontra e ajuda direcioná-lo (ALLIANCE*,* 2013).

Segundo Ken Schwaber (2004), o *Sprint Review Meeting* deve seguir os seguintes critérios:

- O time ágil não deve gastar mais de uma hora para se preparar para o *Sprint* Review Meeting;
- O propósito do *Sprint Review Meeting* é que o time ágil apresente ao *Product Owner* e aos *stakeholders* a funcionalidade feita;

- A funcionalidade que não estiver pronta não pode ser apresentada;
- Artefatos que não são funcionalidades não podem ser apresentados, exceto quando forem usados no suporte para o entendimento da funcionalidade que será apresentada. Artefatos que não podem ser vistos como produtos de trabalho devem ser minimizados para evitar que os *stakeholders* fiquem confusos;
- A funcionalidade pode ser apresentada na estação de trabalho do time ágil e executada por um servidor exclusivo de produção;
- O *Sprint Review Meeting* começa com o time ágil apresentando os objetivos da *Sprint*, o que ele se comprometeu a fazer do *Product Backlog* e o que foi feito dele. Diferentes membros do time ágil podem discutir o que correu bem e o que não correu bem na *Sprint*;
- A maioria do *Sprint Review Meeting* é gasto com o time ágil apresentando a funcionalidade, respondendo as perguntas dos *stakeholders* sobre a apresentação e observando as mudanças desejadas;
- No final da reunião, os *stakeholders* são entrevistados, um por um, para obter suas impressões, algumas mudanças desejadas e suas prioridades;
- O *Product Owner* discute com os *stakeholders* e o time ágil uma possível reorganização do *Product Backlog* baseado nos *feedbacks* obtidos;
- *Stakeholders* são livres para expressar qualquer comentário, observação ou crítica em relação ao incremento da funcionalidade do produto no meio da apresentação;
- Os *stakeholders* podem identificar qualquer nova funcionalidade que ocorrer a eles durante a apresentação e pedir que sejam adicionadas ao *Product Backlog* para priorização. Eles também podem identificar funcionalidades que não foram entregues ou que não foram entregues como esperado e pedir que sejam recolocadas no *Product Backlog* para priorização;
- O *Scrum Master* deve tentar determinar o número de pessoas que podem participar do *Sprint Review Meeting* e configurar a reunião para melhor acomodá-los;

- No final da reunião, o *Scrum Master* anuncia o lugar e a data do próximo *Sprint Review Meeting* ao *Product Owner* e todos os *stakeholders*.

Como resultados do *Sprint Review Meeting*, tem-se o *Product Backlog* revisado que impactará o trabalho que será realizado nas próximas *Sprints* (SCHWABER e SUTHERLAND, 2013).

2.1.5. *Extreme Programming*

Dentre as metodologias ágeis empregadas em desenvolvimento de software, encontra-se o XP (*Extreme Programming*). Segundo Beck (2004), o XP é formado por um conjunto de quatro valores – comunicação, simplicidade, feedback e coragem, sendo todos eles estão diretamente relacionadas a atividades, ações e tarefas do XP.

A comunicação é um item essencial em qualquer processo de desenvolvimento. A fim de obter a comunicação efetiva e eficaz entre todos os membros da equipe e demais *stakeholders*, a XP diz que deve haver colaboração próxima, ainda que informal, entre clientes e desenvolvedores de forma que os requisitos dos projetos estejam claros e sejam rapidamente disseminados entre os membros do time (BECK, 2004).

A simplicidade é alcançada a partir do momento em que a equipe decide realizar apenas as necessidades imediatas, ao invés de especular sobre necessidades futuras, criando, assim, um projeto simples. Caso seja necessário melhorar o projeto, ele deverá passar pelo processo de refatoração (BECK, 2004).

Feedback é um componente de suma importância da *Extreme Programming*. Ele provém de três fontes: o sistema, os usuários e desenvolvedores. A XP faz uso de testes unitários como tática para a obtenção de feedback para o time ágil. À medida que cada classe é desenvolvida, a equipe cria testes unitários para exercitar cada operação de acordo com as funcionalidades especificadas. O grau em que o software implementa o produto, a função e o comportamento do caso de uso é uma forma de feedback (BECK, 2004).

Beck (2004) diz que adotar as práticas da XP exige coragem, pois, diante da enorme pressão nos projetos, há uma tendência para desenvolver pensando no

futuro, achando que a equipe poupará tempo. Na verdade, isso só contribuirá para o fracasso do projeto. Portanto, o time que adotar a XP deve ter coragem para projetar para o hoje, reconhecendo que os requisitos possam sofrer alterações.

Segundo Koscianski e Soares (2007), o XP é a metodologia ideal principalmente para equipes pequenas e médias, desenvolvendo software baseado em requisitos não muito bem claros, onde há grande possibilidade de modificação constante, abordagem incremental e comunicação.

2.1.6. Práticas do XP

O XP traz uma série de boas práticas que, quando aplicadas de maneira correta, visam dar maior agilidade ao desenvolvimento. Algumas das práticas do XP são: cliente presente, propriedade coletiva de código, integração contínua, programação em pares, refatoração, TDD e padrões de códigos (TELES, 2006) (COHN, 2004).

2.1.6.1. Cliente presente

Segundo Beck (2004), um cliente real deve ficar com o time, estar disponível para responder questões, resolver dúvidas ou problemas do negócio e definir prioridades. Sempre que o cliente se faz presente durante a maior parte do tempo dentro dos times de desenvolvimento, o tempo de validação é reduzido, o único modo de saber se o trabalho está correto é através do *feedback* (AMBLER, 2002).

2.1.6.2. Propriedade coletiva de código

Segundo Cohn (2009), propriedade coletiva se refere ao sentimento de posse que todos os membros do time ágil tem sobre todos os artefatos gerados, em especial o código e os testes. Esta prática encoraja cada membro do time ágil a se sentir responsável por todas as partes do programa para que possam trabalhar em qualquer módulo dele. Membros do time ágil mesmo assim vão tender às áreas de sua especialidade ou que preferem trabalhar, porém devem-se certificar que nenhum membro se torne tão especializado que não possa contribuir em outras

áreas e que nenhuma área se torne tão complexa ao ponto de ser compreendida e trabalhada apenas por um membro.

O importante é que todos os desenvolvedores tenham acesso a todos os trechos de código, não sendo dividido em blocos individuais. Todo trecho de código pode ser modificado sem a necessidade de autorização, todos são donos do código.

Segundo Teles (2006), pelo fato de o código ser coletivo, há a necessidade de definição de padrões, para que a compreensão não seja afetada. O uso de padrões durante o desenvolvimento deve ser acompanhado e a fuga do padrão deve ser identificada rapidamente.

2.1.6.3. Integração contínua

Segundo Cohn (2009), integração contínua é a integração do código novo ou alterado para a aplicação o mais rápido possível, seguida de testes que notificaram a ocorrência de algum erro. Através dessa prática, espera-se que o time ágil produza e entregue o código com mais rapidez e que o sistema mantenha a qualidade exigida.

A todo o momento do desenvolvimento, o código gerado deve ser integrado ao restante e testado. Esta prática visa diminuir os impactos das mudanças e novas funcionalidades no código estável.

Sempre que um trecho de código é criado, aconselha-se a executar os testes unitários, porém, mesmo que os desenvolvedores tentem simular todos os cenários para testes, pode acontecer de serem encontrados erros no sistema, sejam eles por erro de negócio, lógica, interpretação e outros fatores (TELES, 2006).

2.1.6.4. Programação em pares

Segundo Cohn (2009), programação em pares é a prática de ter dois membros do time ágil trabalhando juntos na escrita do código. O sentimento de propriedade coletiva é criado e aumentado quando o código é feito em pares e a disciplina de deixar o código mais limpo do que ele estava antes é mais fácil de se atingir quando outro membro do time está sentado ao seu lado.

Quando se desenvolve em pares, há grande chance de se obter um código mais confiável uma vez que, sempre que um programa, o outro revisa o código em tempo real. A divisão dos pares é feita, geralmente, de acordo com a disponibilidade e experiência dos desenvolvedores e, de acordo com Koscianski e Soares (2007), a vantagem desta prática é a transferência de aprendizado entre a dupla, acrescentando, um ao outro, o conhecimento.

2.1.6.5. Refatoração

Segundo Cohn (2009), refatoração é realizar mudanças na estrutura do código sem alterar o seu comportamento. Após vários reparos no código, o sistema pode ficar comprometido e piorar sua performance. Ao utilizar a prática de refatoração, o objetivo é fazer ajustes ou modificações na estrutura do código para melhorar aspectos como coesão e acoplamento e evitar código mal escrito, sem alterar ou até melhorar o comportamento do sistema, em detrimento às mudanças que surgem

Sempre que necessário, o código é refatorado objetivando a melhoria contínua deste. Após a refatoração, o software deve continuar funcionando da mesma maneira que antes, porém, com código mais limpo, organizado e simples. A refatoração depende bastante dos códigos escritos, já que, deve-se fazer uso destes para se certificar do correto funcionamento do sistema após os ajustes executados.

Sempre que o desenvolvedor encontrar determinado trecho de código mal formulado, deve ser feita a refatoração deste trecho, tornando-o mais legível, porém, novamente, sem afetar o funcionamento original do sistema (TELES, 2006).

2.1.6.6. Desenvolvimento orientado a testes (TDD)

Segundo Cohn (2009), a prática de TDD visa melhorar a qualidade do código através de uma cobertura de testes. O teste que, antes era feito depois do código estar pronto, agora é feito antes de construí-lo, sendo gerado a partir do teste.

Qualquer aspecto ou função do programa que não tenha um teste automatizado simplesmente não existe. Os programadores escrevem teste de

unidade para que sua confiança no programa possa se tornar parte da aplicação, os clientes escrevem testes de funcionalidades para que a sua confiança também faça parte da aplicação e o resultado é um programa mais confiável (BECK, 2004).

Cohn (2004) afirma que TDD garante que o código fique mais enxuto e testável, melhorando também sua manutenção, que agora é realizada desde o momento que o código é criado.

2.1.6.7. Padrões de código

Para que um time tenha propriedade coletiva em seu próprio código produzido, é importante que ele tenha e siga padrões de escrita de código, pois esses padrões de código estabelecem as principais regras e consensos entre a equipe sobre como o código deve ser escrito, por exemplo como as variáveis e métodos serão nomeados e como as linhas serão formatadas (COHN, 2004).

Se um time ágil tem todos seus programadores trabalhando em tantas partes diferentes do sistema, trocando de duplas e refatorando o código constantemente, é necessário definir padrões de código, devendo exigir a menor quantidade de trabalho possível e enfatizar a comunicação, evitando código duplicado e sendo adotado voluntariamente (BECK, 2004).

2.1.7. Arquitetura do sistema no XP

A arquitetura é um fator tão importante em projetos XP quanto em outro projeto de software, sendo parte dela capturada pela metáfora do sistema (BECK, 2004).

As metáforas são usadas com o intuito de facilitar ao máximo a compreensão do sistema. As coisas funcionam e se relacionam “simulando” componentes do mundo real. Desta maneira, segundo Teles (2006) é possível “transmitir ideias complexas de forma simples, através de linguagem comum” dos clientes aos desenvolvedores e vice-versa.

Beck (2004) afirma que a metáfora na XP “substitui muito daquilo que outras pessoas chamam de Arquitetura”, afinal, preserva o objetivo da arquitetura: dar a todos uma história coerente para que possam trabalhar, de fácil compartilhamento

entre negócio e desenvolvimento, alcançando uma arquitetura fácil de elaborar e comunicar.

É importante ressaltar que a metáfora serve para facilitar a compreensão do sistema, sendo apenas, como o próprio nome já diz, metáforas. Sendo assim, ao dizer que o sistema “é como uma planilha de cálculo” não significa dizer que de fato o sistema “é uma planilha de cálculo” (BECK, 2004). Beck (2004) também deixa claro que o desenvolvimento se faz impossível partindo de apenas uma metáfora, já que desta maneira não se teria detalhes suficientes e seria um risco muito grande caso ela esteja errada, a não ser que o cliente se sinta à vontade falando sobre o sistema com esta metáfora e que houvesse refatoração contínua para reafirmar o entendimento desta na prática.

Em colaboração, Ambler (2002) diz que a arquitetura do sistema deve surgir naturalmente com implementação das *user stories* mais prioritárias, sendo essas histórias derivadas da metáfora criada para o sistema.

2.1.8. Atividades do time ágil

Considerando o que foi dito nas seções anteriores, o Quadro 1 mostra todas as atividades que o time de ágil deve exercer, relacionando cada uma delas com um evento do *Scrum* a qual pertence e se há relacionamento com outros papéis.

Quadro 1. Resumo do papel do time ágil (O autor).

Cerimônias do Scrum	**Relacionamentos**		**Atividades**
	Scrum Master	*Product Owner*	
Sprint Planning Meeting	Sim	Sim	- Selecionar os itens do *Product Backlog* que se compromete a fazer; - Fazer sugestões ao *Product Owner* quanto ao *Product Backlog*;

Cerimônias do Scrum	**Relacionamentos**		**Atividades**
	Scrum Master	*Product Owner*	
			- Determinar quanto do *Product Backlog* que o *Product Owner* quer tentará ser feito durante a *Sprint*; - Questionar o *Product Owner* sobre qualquer dúvida referente ao *Product Backlog*; - Preparar o *Sprint Backlog* com as tarefas estimadas e seus respectivos responsáveis definidos.
Daily Scrum Meeting	Sim	Não	- Relatar o andamento do trabalho.
Sprint	Sim	Sim	- Procurar informações e apoio de fora do time de desenvolvimento, *Product Owner* e *Scrum Master*, para realizar o trabalho; - Requerer o cancelamento da *Sprint* ao *Scrum Master*; - Requerer ao *Product Owner* o acréscimo de itens do *Product Backlog* para o *Sprint Backlog* da *Sprint* atual; - Requerer ao *Product Owner* a remoção de itens do *Sprint Backlog* da *Sprint* atual; - Refinar e atualizar o *Sprint Backlog* diariamente;

Cerimônias do Scrum	**Relacionamentos**		**Atividades**
	Scrum Master	*Product Owner*	
			- Analisar o *Sprint Backlog* para melhor compreensão; - Escrever o código; - Escrever testes enquanto o código está sendo escrito; - Escrever métodos necessários para suportar os testes que foram ou serão escritos; - Testar o código contra os testes escritos; - Fazer check-in de código novo no repositório compartilhado; - Fazer check-in de novos casos de testes no repositório compartilhado; - Integração de testes em um ambiente de teste contínuo; - Refatorar o código quando necessário; - Realizar as melhorias priorizadas no *Sprint Retrospective Meeting*.
Sprint Retrospective Meeting	Sim	Sim (Opcional)	- Relatar como foi o andamento da *Sprint*; - Identificar o que pode ser melhorado para a próxima *Sprint*; - Priorizar as potenciais melhorias.

Cerimônias do Scrum	**Relacionamentos**		**Atividades**
	Scrum Master	*Product Owner*	
Sprint Review Meeting	Sim	Sim	- Apresentar o que foi feito na última *Sprint* ao *Product Owner* e aos *stakeholders*; - Entregar o incremento do produto feito; - Responder aos questionamentos do *Product Owner* e dos *stakeholders*; - Observar as mudanças desejadas pelo *Product Owner* e pelos *stakeholders;* - Discutir com o *Product Owner* e com os *stakeholders* sobre possíveis mudanças no *Product Backlog.*

Concluído o detalhamento dos métodos ágeis *Scrum* e XP com foco no time ágil e mapeadas as atividades desse time, nas próximas seções são apresentados os conceitos de competência no trabalho.

2.2. COMPETÊNCIAS

As mudanças nas sociedades, principalmente nas relações humanas, impactam de forma direta a rotina das empresas, onde adaptações constantes nas práticas de gerenciamento de pessoas são necessárias.

O interesse sobre competências é uma dessas mudanças. Picarelli e Wood (1999) afirmam que desde os anos 80, executivos e pesquisadores notaram as

competências e capacidades como uma fonte importante e sustentável de vantagem competitiva. Afirmam também que apesar do discurso de vários dirigentes já terem institucionalizado este conceito na prática não é utilizado adequadamente:

> A maioria ainda subestima o impacto que as competências individuais podem ter sobre o resultado dos negócios e costumam subordinar as pessoas à tecnologia ou aos sistemas de trabalho, esquecendo que são as pessoas que desenvolvem e operam novas tecnologias e novos sistemas (PICARELLI e WOOD, 1999, p. 45).

Eles apontam as razões para isto ao mencionar ser difícil lidar com pessoas, contribuindo para a tendência de desqualificar o que não se conhece ou tem controle.

Em contrapartida, para Gramigna (2002) competências e sua gestão não podem ser mais taxadas apenas como um modismo, pois considerando resultados de mais de 25 anos de estudos já realizados em organizações, é possível comprovar a sua eficiência.

Segundo Resende (2003), competências podem ser vistas de diferentes maneiras: o uso de competências como uma abordagem para gerenciar as pessoas pode ser considerado como uma quebra de paradigma, pois o salário das pessoas deixa ser pago por tempo de casa, ou por escolaridade, e passa a acontecer de acordo com as competências que estas pessoas tem. Elas também podem ser tratadas como uma era ou um movimento, pois é uma fase de referência na evolução da sociedade e das empresas e se expandem em diversas novas direções e em novas e diferentes áreas de atuação, possibilitando um leque de alternativas para aplicações.

Resende (2003) também menciona a competitividade tendo um papel significante na valorização da abordagem por competências, pois a busca por ela afeta em tudo que se relaciona ao mundo dos negócios, desde o desenvolvimento de novas tecnologias até os empregos e profissões. Sendo assim, a grande procura por competitividade nas empresas cria a necessidade de captar o máximo de pessoas, que por sua vez, devem procurar se tornar os melhores recursos humanos, transferindo a competitividade também às pessoas o que aumenta o interesse das empresas por esta abordagem para gerenciar esses recursos.

Fleury e Fleury (2004, p. 78) ao tratar a construção de um modelo de gestão de pessoas segundo o conceito de competências postulam:

> A década de 90 com seus desafios de crescente competitividade e globalização das atividades, levou ao alinhamento definitivo das políticas de gestão de recursos humanos às estratégias empresariais incorporando à prática organizacional o conceito de competência, como base do modelo para se gerenciarem pessoas.

Cabe ressaltar também que não apenas os funcionários ou candidatos a funcionário das empresas precisam se alertar quanto às competências. Profissionais liberais ou autônomos também estão susceptíveis a esta pressão e começam a se preocupar em construir seu perfil através da abordagem de competências (FERRARINI, 2006).

Le Boterf (2003) diz que ao se tratar de competência encontram-se diversas definições e conceitos. Segundo Klimnik e Sant'anna (2006) há duas vertentes predominantes: a perspectiva anglo-saxônica que define competências tomando como referência o mercado de trabalho e enfatizando fatores ou aspectos ligados a descritores de desempenho requeridos pelas organizações; e a francesa, que enfatiza a vinculação entre trabalho e educação, indicando as competências como uma resultante de processos sistemáticos de aprendizagem.

De acordo com Fleury e Fleury (2001), McClelland, em 1973, iniciou o debate sobre competência entre psicólogos e administradores nos Estados Unidos, ao publicar um artigo chamado *Testing for Competence rather than Intelligence*. Ainda contemplando o que foi dito por esses autores, o debate francês a respeito de competência nasceu também nos anos 70, tratando justamente a partir do questionamento do conceito de qualificação técnica e do processo de formação profissional.

Dentre os vários conceitos encontrados, Resende (2003) define competência como a transformação de conhecimentos, aptidões, habilidades, interesse, vontade, etc. em resultados práticos, onde ter conhecimento e experiência e não saber aplicá-los em favor de um objetivo significa não ser competente.

Já uma definição voltada a um nível organizacional, Hamel e Prahalad (1995) apud Coelho Neto (2005) mencionam que competência é um conjunto de

habilidades e tecnologias, e não uma única habilidade isolada, sendo uma competência específica de uma organização representada pela soma de todos os conjuntos de habilidades tanto em nível pessoal quanto da unidade organizacional.

Zarifian (2001) utiliza uma definição que envolve várias dimensões e pondera que o conceito de competência vai além do de qualificação, dizendo ser a competência o 'tomar iniciativa' e o 'assumir responsabilidade' do indivíduo diante das situações profissionais que ele encontra, o entendimento prático de situações que se baseiam em conhecimentos adquiridos e os transforma na medida que aumenta a diversidade das situações, a faculdade de mobilizar vários atores em torno das mesmas situações e de fazê-los compartilharem as implicações de suas ações e assumirem áreas de corresponsabilidade.

Para Le Boterf (2003) a competência profissional é saber administrar uma situação profissional complexa, sendo que este saber que caracteriza competência implica em um saber agir responsável, saber mobilizar saberes e habilidades em um contexto profissional, saber integrar ou combinar conhecimentos e recursos múltiplos, saber transpor conhecimentos, recursos ou habilidades para novas situações, saber aprender a aprender e, ainda, saber envolver-se, em um contexto profissional.

Fleury e Fleury (2001) conceituam competência como "um saber agir responsável e reconhecido, que implica mobilizar, integrar, transferir conhecimentos, recursos, habilidades, que agregam valor econômico à organização e valor social ao indivíduo". Contribuindo para essa definição, a Figura 2 demonstra a associação dos saberes e ações mencionados no conceito com competência.

Figura 2. Competências como fonte de valor para o indivíduo e para a organização.
Fonte: (FLEURY e FLEURY, 2001)

Um dos conceitos mais aceitos é o definido por Parry (1996, p. 50) que traz competência sendo:

> Um agrupamento de conhecimentos, habilidades e atitudes correlacionados, que afeta parte considerável da atividade de alguém, que se relaciona com o desempenho no trabalho, que pode ser medido segundo padrões adequados, e que pode ser melhorado por meio de treinamento e desenvolvimento.

Em consonância com este conceito, Durand (2006) propõe competência a partir do relacionamento entre conhecimento (*Connaissance*), habilidades (*Pratiques*) e atitudes (*Attitudes*), sendo:

- Conhecimento: conjunto de saberes que uma pessoa adquiriu na sua formação;
- Habilidades: experiência adquirida pela prática;
- Atitudes: fazer com que as coisas realmente aconteçam.

Contribuindo com a visão de Durand (2006), Fleury e Fleury (2001) e Parry (1996), Leme (2006) também conceituam competência a partir desses três pilares, mas separando-a em competências técnicas e comportamentais, onde:

- Competências técnicas abordam conhecimentos e habilidades específicas sobre o trabalho que deve ser realizado pelo servidor nas diferentes unidades de trabalho, como por exemplo o fato de conhecer leis e normas e ter capacidade de utilizar ferramentas;
- Competências comportamentais envolvem as atitudes e os valores incorporados ao desempenho do servidor, como por exemplo o fato de ter iniciativa e criatividade.

Para melhor demonstrar este conceito, o Quadro 2 ilustra como se dá o uso das definições de conhecimentos, habilidades e atitudes (CHA) com a perspectiva de competências técnicas e comportamentais.

Segundo Leme (2006), selecionar e avaliar pessoas a partir de competências técnicas e comportamentais é uma maneira de atingir o equilíbrio de forma sustentada e alinhada à estratégia da organização, recrutando pessoas que poderão não só trazer resultados à empresa, mas também gerar crescimento a ela.

Quadro 2. Modelo CHA x Competências técnicas e comportamentais (Adaptado de (LEME, 2006)).

Competência Técnica		**Competência Comportamental**
Conhecimento	Habilidade	Atitude
Saber a técnica	Saber executar a técnica	Querer fazer

No desenvolvimento deste trabalho, foi utilizado o conceito de competência definido por Durand (2006), Leme (2006) e Parry (1996), que dispõe a dinâmica das três dimensões (conhecimento, habilidade e atitude), unido com a distinção defendida por Leme (2006) de competências técnicas e comportamentais, tornando o trabalho responsável por definir esses dois tipos de competências para um time ágil.

3. METODOLOGIA DE PESQUISA

3.1. TIPOLOGIA DA PESQUISA

A metodologia de pesquisa utilizada se divide em três pontos de vista que colaboram e consideram os objetivos definidos e a problemática deste trabalho. Os pontos de vista da metodologia de pesquisa são:

- Ponto de vista de sua natureza:

Este trabalho fez uso de uma pesquisa aplicada, pois objetiva gerar conhecimentos para aplicações práticas em detrimento dos problemas específicos encontrados.

- Ponto de vista dos objetivos:

Este trabalho fez uso de uma pesquisa exploratória, pois envolveu um levantamento bibliográfico e foi realizado nas próximas etapas do trabalho entrevistas com pessoas detentoras de experiências práticas com o problema pesquisado.

- Ponto de vista dos procedimentos técnicos:

Este trabalho fez uso de uma pesquisa bibliográfica, pois foi elaborado a partir do que já fui publicado em livros, artigos, periódicos, apenas materiais que já receberam um tratamento analítico.

Tendo o problema, os objetivos e os tipos de pesquisa definidos, foi feita uma revisão bibliográfica e utiliza-se a técnica de revisão sistemática, melhor detalhada na Seção 3.2.

3.2. REVISÃO SISTEMÁTICA

A revisão bibliográfica foi realizada com o apoio da técnica de revisão sistemática.

Para iniciar uma revisão sistemática, é necessário seguir um protocolo que contém todas as informações iniciais que essa revisão deve ter. Esse protocolo está disponível no Apêndice I deste documento.

As buscas foram executadas utilizando as máquinas de busca das bibliotecas digitais *ACM Digital Library*, *IEEEXplore* e *ScienceDirect*. A primeira biblioteca a ser executada a busca foi *IEEEXplore*, onde foi testada e reformulada a *string* de busca até que estivesse retornando cerda de 50 a 150 de trabalhos. Já na *ACM Digital Library*, a *string* de busca foi apenas adaptada à maquina de busca dessa biblioteca.

A busca executada na biblioteca *ScienceDirect* foi realizada por último e, por ter uma máquina de busca diferente das outras bibliotecas utilizadas, foi feito mesmo procedimento com a *IEEXplore* até que a *string* de busca retornasse o mesmo limite de quantidade de trabalhos.

Foram retornadas 364 referências, sendo 47 para *ACM Digital Library*, 267 para *IEEEXplore*, sendo considerado apenas as 75 referências mais relevantes dessa biblioteca, e 50 para *ScienceDirect* contabilizando 172 referências no total. Para ajudar na captura, análise e tratamento dos trabalhos retornados foram utilizados duas ferramentas: *Zotero* e *StArt*. *Zotero* é uma ferramenta gratuita que serve como um gerenciador de referências, permitindo coletar, organizar, citar e compartilhar suas fontes de pesquisa. *StArt* (*State of the Art through Systematic Review*) é uma ferramenta gratuita desenvolvida pelo Laboratório de Pesquisa em Engenharia de Software (LAPES) pertencente ao Departamento de Computação da Universidade Federal de São Carlos (DC/UFSCar) e tem como objetivo ajudar o pesquisador dando suporte para a aplicação correta da técnica de revisão sistemática.

Das 172 referências consideradas, foram encontrados 5 itens duplicados. Esses itens foram removidos, mantendo apenas o primeiro importado no gerenciador.

A primeira avaliação feita para filtrar as referências foi realizada na própria *string* de busca, restringindo as palavras-chave no título dos trabalhos. Em seguida, a segunda avaliação foi realizada diretamente no corpo dos trabalhos a partir da leitura do resumo e verificação dos índices dos documentos, resultando em 59 referências aceitas para a próxima verificação. A terceira e última avaliação foi dada através da leitura parcial dos itens aceitos na verificação anterior e apenas 15 das 59 referências aceitas foram selecionadas para compor a revisão sistemática. Todas as avaliações foram feitas a partir dos critérios de seleção definidos.

No Gráfico 1 é possível verificar um resumo do que foi obtido nessa etapa de avaliação da revisão sistemática.

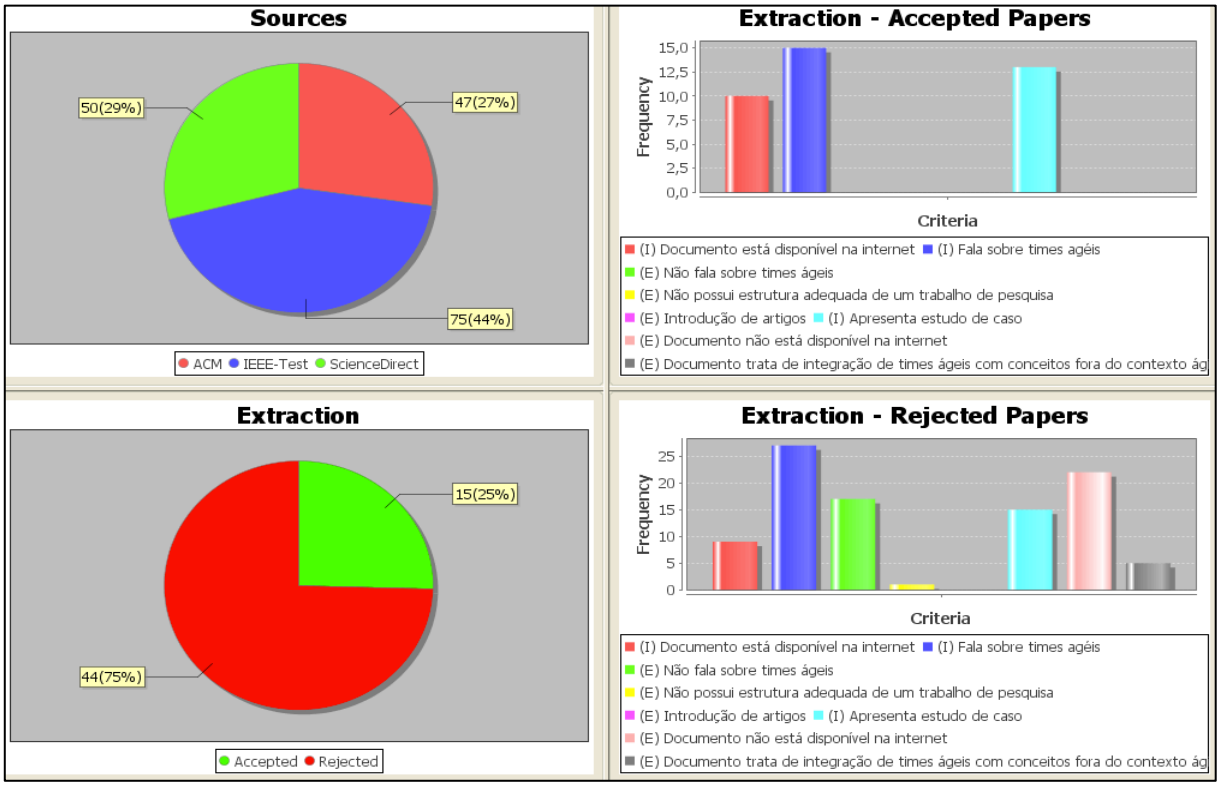


Gráfico 1. Resumo das avaliações (O autor).

Após a leitura completa das referências selecionadas para a revisão sistemática, foi possível encontrar 18 características para times ágeis. Este resultado está ilustrado no Gráfico 2. Essas características foram utilizadas como uma das bases para definir um conjunto de competências para um time ágil.

Junto com as características, foi analisado também o quanto cada uma delas está fundamentada nas referências que as mencionam. Essa verificação foi feita tanto para ajudar na solução do problema que o trabalho se propõe a tratar quanto para evidenciar a contribuição que cada referência traz para o estado da arte encontrado. Essa análise classifica cada característica como citada, explicada e evidente em estudo de caso, podendo ser uma, duas ou as três classificações simultaneamente, conforme demonstrado no Quadro 11 disponível no Apêndice II deste documento.

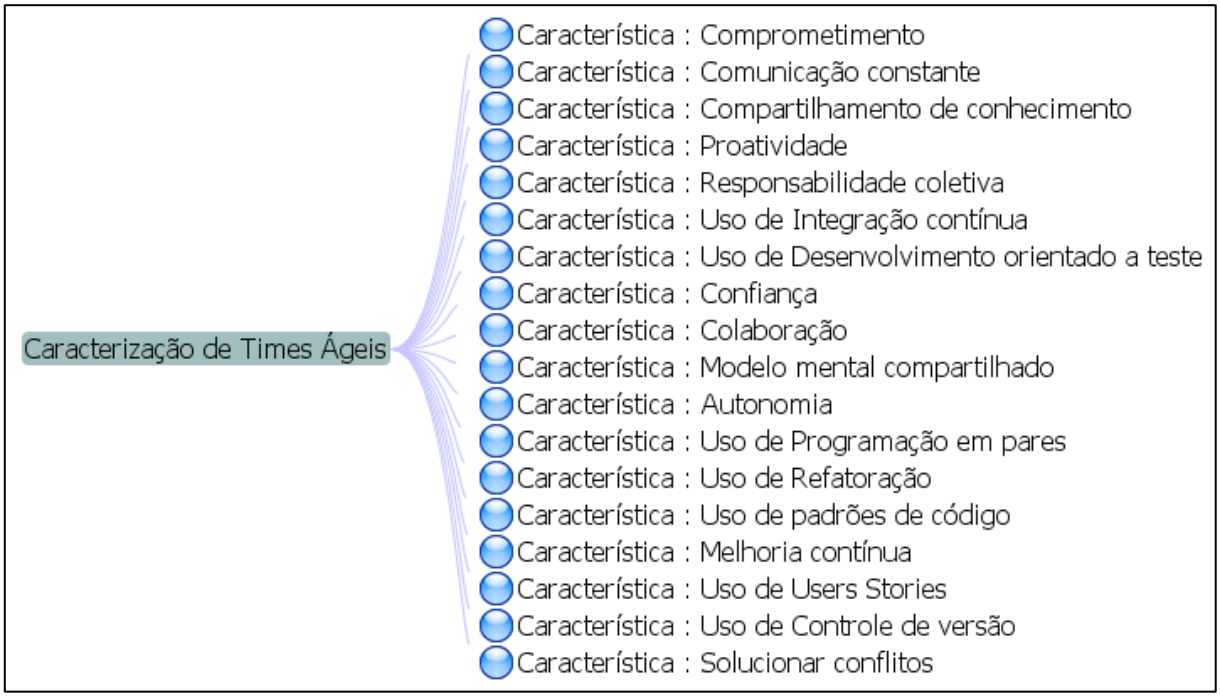


Gráfico 2. Características de times ágeis (O autor).

A partir da revisão bibliográfica realizada e seus resultados obtidos, foi possível desenvolver o Referencial Teórico deste trabalho que pode ser encontrado na Seção 2.

Com base na revisão bibliográfica (com as características encontradas na revisão sistemática), no referencial teórico e no mapeamento das atividades e relacionamentos do time ágil feito, foi derivado e elaborado um conjunto preliminar de competências para um time ágil. Este conjunto estabelecido dispõe de competências comportamentais e técnicas para este papel específico que se encontram nos Quadros 12 e 13, presentes no Apêndice III.

Proposto esse conjunto inicial de competências, foi avaliado a importância dessas competências e para isso foi desenvolvido um instrumento de pesquisa, sendo esse procedimento descrito e detalhado na Seção 3.3.

3.3. INSTRUMENTO DE PESQUISA

Nesta parte do trabalho, é descrita como foi feita a elaboração do instrumento de pesquisa, sua aplicação, validação e análises pertinentes com base no conjunto de competências proposto.

3.3.1. Elaboração do instrumento de pesquisa

Esta etapa do trabalho foi destinada para a construção de um questionário, instrumento principal para o levantamento de dados por amostragem. Levantamento de dados, tradução do termo inglês *survey*, é definido por Fink & Kosecoff (1985) como método para coletar informações de pessoas a partir de suas ideias, sentimentos, planos, crenças, assim como origem social, educacional e financeira. O instrumento usado em um levantamento de dados é o questionário que pode ser definido como um conjunto de perguntas sobre um tópico específico que não avalia ou testa a habilidade do respondente, mas busca sua opinião, seus interesses, aspectos de personalidade e informação bibliográfica (Yaremko, et. al., 1986).

Segundo Hartmut Günther (1999), na elaboração de um questionário para um levantamento de dados deve-se primeiro questionar qual o objetivo da pesquisa em termos dos conceitos a serem pesquisados e da população alvo. Em colaboração com Hartmut Günther (1999), Schuman e Kalton (1985) define na Figura 3 os estágios principais de um *survey* que podem ser usados como guia para a construção de um.

O objetivo da pesquisa desenvolvida neste trabalho é validar o conjunto de competências comportamentais e técnicas de um time ágil, elaborado com base nas características de times ágeis encontradas na revisão sistemática e em suas atividades presentes no referencial teórico.

Com o objetivo de pesquisa definido, é possível determinar os conceitos e a população alvo investigados pelo questionário e, a partir de cada um deles, derivar os itens, perguntas concretas a serem apresentadas, e a amostra, pessoas específicas que respondem as perguntas, respectivamente (GÜNTHER, 1999).

O conceito a ser explorado pelo questionário é o grau de importância dado a cada competência e os itens do questionário são as próprias competências. Já a população alvo são as pessoas que utilizaram, utilizam ou estudam metodologias ágeis de desenvolvimento de software, configurando como amostras dessa população os professores e estudantes de cursos relacionados à Engenharia de Software e profissionais com experiência em metodologias ágeis tanto da iniciativa privada quanto do setor público.

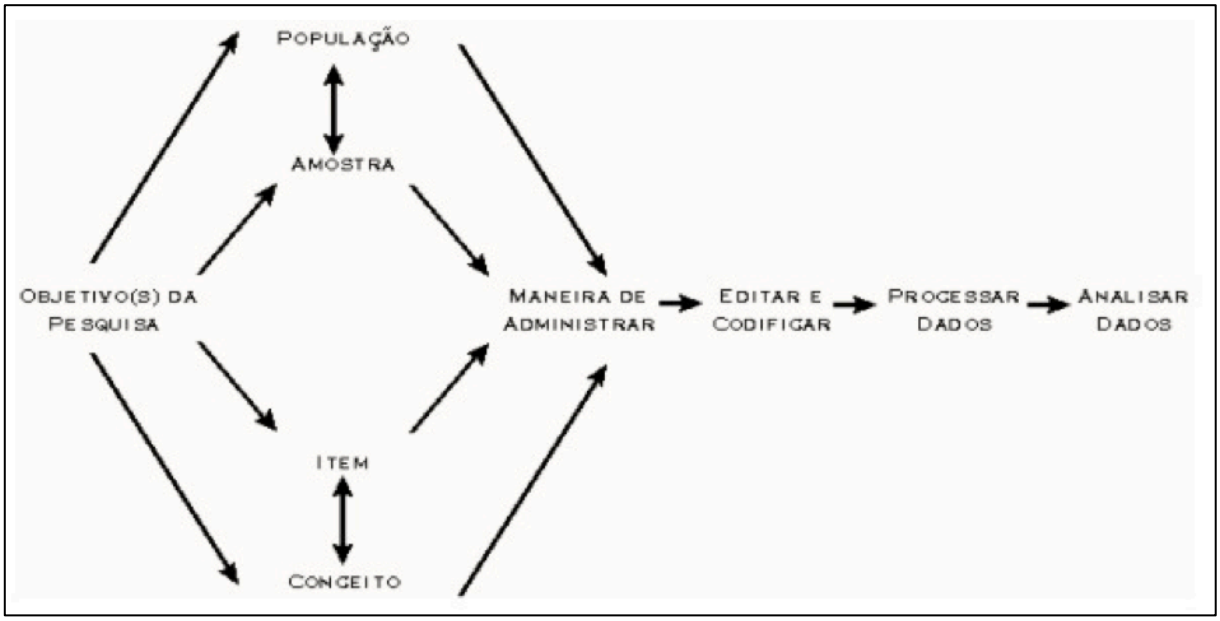


Figura 3. Estágios principais de um *survey*. Fonte: (SCHUMAN e KALTON, 1985).

Para medir o grau de importância de cada competência, utilizou-se a escala *Likert* que é uma mensuração mais utilizada nas ciências sociais, em especial para levantamentos de atitudes, opiniões e avaliações (GÜNTHER, 1999). A escala definida é disposta em cinco alternativas, sendo elas:

1. Nada importante;
2. Pouco importante;
3. Mediamente importante;
4. Muito importante;
5. Extremamente importante.

A escolha por uma escala com um número ímpar de alternativas permite ao respondente não se comprometer em alguma pergunta que ele não possui tanto conhecimento podendo marcar um ponto neutro, por isso essa foi a escala adotada (GÜNTHER, 1999).

O questionário foi feito através da ferramenta Google Drive que ajuda na construção e disponibilização do instrumento de pesquisa e, principalmente, na coleta das respostas, possibilitando uma visão resumida de todos os dados obtidos.

Em concordância com o que foi definido nesta etapa, a primeira versão do questionário foi concebida e pode ser encontrada no Apêndice IV deste trabalho. A próxima etapa foi a validação do questionário que será descrita e detalhada na próxima seção.

3.3.2. Validação do instrumento de pesquisa

A etapa de validação do instrumento de pesquisa elaborado consistiu em uma avaliação crítica feita por profissionais atuantes no setor público e privado que utilizam ou já utilizaram desenvolvimento ágil de software em projetos que eles participaram e de professores do curso de Engenharia de Software da Universidade de Brasília por meio de entrevistas. Essa avaliação persistiu na opinião do entrevistado sobre os seguintes critérios:

- Estrutura (organização e tamanho do questionário);
- Semântica (compreensão e interpretação do questionário);
- Conteúdo (contexto no qual a pesquisa está inserida).

A partir desses critérios, a entrevista se tornou mais objetiva, pois quanto mais rápida e simples ela for, maior a probabilidade do entrevistado participar dessa validação.

A validação se restringiu a oito entrevistados, sendo quatro professores, dois profissionais da iniciativa privada e dois profissionais do setor público. Os profissionais que forem entrevistados utilizam ou utilizaram metodologias ágeis ao longo de sua carreira em desenvolvimento de software.

O objetivo dessa validação foi verificar se o instrumento de pesquisa está consistente e apto com base na opinião dos entrevistados para coletar as informações que ele se propõe gerar. Vale ressaltar que essa etapa não houve respostas dos entrevistados ao questionário, apenas suas sugestões de melhoria e mudanças.

3.3.2.1. Área Acadêmica

Dentre os professores que participaram dessa validação, cabe ressaltar que todos ministram ou já ministraram disciplinas no curso de Engenharia de Software da Universidade de Brasília (UnB), onde desenvolvem pesquisas por meio de artigos científicos e projetos de pesquisa em parceria com a iniciativa privada e o setor público e também são participantes de comunidades ágeis.

As entrevistas envolvendo esses profissionais foram realizadas entre os dias 03/04/2014 e 09/04/2014, sendo coletadas suas observações críticas sobre o que poderia ser melhorado e mudado no questionário. Os professores ressaltaram as seguintes observações:

- O questionário estava muito extenso, mas completo e necessário;
- Deve ser ressaltado que as competências devem circular e se propagar entre o time por inteiro para que não ocorra a dependência em apenas uma pessoa que detém uma determinada competência;
- As competências técnicas “Dominar linguagens de programação”, “Dominar a prática de Integração contínua de software”, “Dominar a prática de Refatoração de código”, “Dominar práticas de desenvolvimento da Arquitetura do sistema”, “Dominar práticas de desenvolvimento do banco de dados do sistema” e “Dominar práticas e padrões de gerência de configuração de software” poderiam ser revistas ou até retiradas, pois são comuns a qualquer time que desenvolve software independente da metodologia utilizada;
- Deveria ser realçado o caráter diário da transferência e compartilhamento do conhecimento entre o time;
- Na caracterização da amostra poderia ser questionado se a experiência que o respondente teve com o desenvolvimento ágil de software foi satisfatória;
- Redefinir ou repensar o público alvo da pesquisa para captar respostas mais maduras e experientes;
- Omitir do formulário o fato da pesquisa fazer parte de um trabalho de conclusão de curso, pois há leitores que podem desvalorizar o trabalho por conta disso;

- Informar o conceito de competência utilizado de competência no trabalho e sua referência;
- Deixar explícito a quantidade de perguntas em cada parte do questionário ao leitor para orientá-lo antes de ler e responder;
- Criar cabeçalhos de introdução para competências comportamentais e técnicas junto com seus conceitos;
- O conjunto de competências aparentou ser genérico a qualquer time independente de metodologia;
- Especificar nas competências técnicas que o termo “dominar” significa ter o conhecimento e habilidade para determinada prática ou rever se é melhor dividir em conhecimento e habilidade;
- As competências comportamentais ficaram mais claras que as técnicas;
- Todas as competências elaboradas aparentam ser importantes, pois as palavras utilizadas na descrição das competências como importantes, constantes, evidentes podem induzir o leitor a achá-la relevante.

Após o questionário ser avaliado por profissionais atuantes na área acadêmica, o instrumento de pesquisa foi validado por profissionais atuantes no setor público, sendo melhor descrito na Seção 3.3.2.2.

3.3.2.2. Setor público

Dentre os profissionais atuantes no setor público, é válido ressaltar os órgãos envolvidos nessas entrevistas como o Conselho Nacional de Justiça (CNJ) e a empresa estatal Terracap. Não obstante, engrandece a pesquisa ao saber que todos esses entrevistados já utilizaram metodologias ágeis ao longo de sua carreira, alguns utilizam atualmente também, tendo essa experiência exercida em locais como o Conselho Federal de Contabilidade (CFC) e a Companhia Nacional de Abastecimento (CONAB).

Os perfis dos profissionais participantes dessa validação são perfis como analistas de sistemas, gerentes de projeto e chefe de seções, o que enriquece ainda mais a validação com opiniões de cargos importantes.

As entrevistas envolvendo esses profissionais foram realizadas entre os dias 11/04/2014 e 15/04/2014, onde foram ressaltadas as seguintes observações:

- Assim como foi opinado pelos professores, o questionário está extenso, porém completo e abrangente;
- O trabalho está muito bom e o conceito de competência utilizado foi relevante para a pesquisa;
- Unificar a competência “Capacidade de compartilhar e transferir o conhecimento obtido” com a competência “Disposição para ajudar o time ou membros do time sempre que for necessário”;
- Unificar a competência “Capacidade de tomar decisões diante de situações de incerteza” com a competência “Responsabilidade pelas atividades e tarefas do time com um todo e por tudo que foi produzido por ele”;
- A competência técnica “Conhecer a instituição em que trabalha para desempenhar seu papel” deveria ser considerada como comportamental e se fundir com à competência comportamental “Facilidade de comunicação com todos os níveis da organização”;
- A competência técnica “Dominar técnicas de programação” aparentou ser muito genérica e poderia ser retirada, já que está sendo especializada nas competências posteriores;
- O tema do trabalho é muito interessante e proverá resultados relevantes ao contexto que está inserido;
- O trabalho, a partir das competências elaboradas e validadas, poderia gerar uma ótima ferramenta de avaliação de times ágeis;
- A maior parte das competências necessárias para um time ágil estão listadas e descritas no trabalho, porém poderia ser incluído alguma competência relacionada à capacidade do time em estimar o esforço necessário para realizar uma atividade ou user story de maneira satisfatória.

Após o questionário ser avaliado por profissionais do setor público, o instrumento de pesquisa foi validado por profissionais atuantes na iniciativa privada, sendo melhor descrito na Seção 3.3.2.3.

3.3.2.3. Iniciativa privada

Dentre os profissionais atuantes na iniciativa privada, é válido ressaltar as empresas envolvidas nessas entrevistas como a SEA Tecnologia e o Instituto Eldorado de Pesquisa.

Os perfis dos profissionais participantes dessa validação são perfis como analistas de sistemas e diretor de operações, o que enriquece ainda mais a validação com opiniões de cargos importantes. A entrevista foi realizada entre os dias 05/04/2014 a 11/04/2014, onde foram ressaltadas as seguintes observações:

- O questionário estava excelente, principalmente quanto a parte das competências comportamentais, porém poderia ser incluído na parte técnica algo referente à capacidade de negociação de escopo para que o time possa trabalhar sem pressão ou desespero;
- A competência técnica “Conhecer a instituição em que trabalha para desempenhar seu papel” precisa ser melhor explicada, pois aparentou ter mais relação com aspectos comportamentais;
- A competência “Facilidade de comunicação com todos os níveis da organização” menciona em sua descrição os *stakeholders* pertencendo à organização, porém eles podem pertencer a uma outra organização, que pode ser ou não envolvida com o projeto;
- A competência “Flexibilidade e equilíbrio para gerenciar os conflitos do time” fala em sua descrição sobre solucionar conflitos, porém nem sempre é possível solucioná-los, sendo mais importante saber lidar com eles;
- As competências técnicas como, por exemplo, “Dominar a prática de Integração contínua”, “Dominar a prática de Refatoração de código” e “Dominar a prática de User Stories” poderiam ser melhor descritas com a definição da prática em questão;
- O questionário ficou muito longo e realizar alguns cortes podem ajudar;
- As descrições das competências precisam ser diretas e curtas com informações relevantes apenas ao leitor, sem a necessidade de padronizá-las e com termos específicos do trabalho;

- Pensar em algum benefício para que o leitor se sinta estimulado a contribuir com sua opinião como um brinde ou presente.

Encerradas as entrevistas para validação do instrumento de pesquisa, foram analisadas as observações feitas pelos entrevistados e realizados ajustes no questionário.

3.3.3. Ajuste do instrumento de pesquisa

Com tudo o que foi levantado na etapa de validação e com base nas observações feitas pelos entrevistados, as críticas e elogios foram analisados e ponderou-se o que era considerado como relevante para a melhor coesão e interpretação do questionário. Diante disso e junto aos orientadores deste trabalho, foram realizadas as seguintes alterações e considerações no instrumento de pesquisa:

- Foram feitos cortes e simplificação do texto introdutório do questionário e da descrição das competências;
- Foi incluído a competência técnica relacionada à capacidade de negociação de escopo (CT15);
- A competência técnica “Conhecer a instituição em que trabalha para desempenhar seu papel“ foi melhor descrita para ressaltar a necessidade de se conhecer os padrões e normas que o time ágil utiliza e as pessoas com quem lida (CT1);
- Todas as descrições foram refeitas para ficarem mais claras e objetivas;
- Foi retirada a competência técnica voltada para técnicas de programação, pois estava especializada por competências posteriores;
- Foi incluída a competência técnica “Dominar o uso de padrões de codificação” (CT6);
- Foi incluída a competência técnica referente à capacidade do time de estimar esforço de acordo com suas limitações para que seja realista e satisfatória (CT16);

- Mesmo genéricas, as competências técnicas “Dominar linguagens de programação”, “Dominar a prática de Refatoração de código”, “Dominar práticas de desenvolvimento da Arquitetura do sistema”, “Dominar práticas de desenvolvimento do banco de dados do sistema” e “Dominar práticas e padrões de gerência de configuração de software” foram mantidas no questionário, pois o intuito era captar todas as competências necessárias para um time ágil independente de sua generalidade;
- Foi adicionado à descrição da competência “Capacidade de compartilhar e transferir o conhecimento obtido” o caráter diário da transferência e compartilhamento do conhecimento entre o time ágil (CC1);
- A competência “Zelo e empenho pelo cumprimento das atividades e obrigações do time” foi reformulada para “Capacidade de cumprir com seus compromissos” (CC6);
- Na caracterização da amostra foi incluída uma pergunta referente a satisfação do respondente quanto a sua experiência com desenvolvimento ágil de software;
- O público alvo foi ajustado, se restringindo às pessoas que utilizaram ou utilizam metodologias ágeis ou estudam assuntos ligados a isso;
- Foi omitido do questionário o fato da pesquisa estar vinculada a um trabalho de conclusão de curso;
- Não foi informado a quantidade de perguntas em cada parte do questionário para que o leitor não desistisse de respondê-lo;
- Foram criados os cabeçalhos de introdução e junto a eles foram colocadas breves descrições sobre o que se trata cada tipo de competência de acordo com o conceito utilizado;
- O fato das competências estarem diluídas entre todos os membros do time já está implícito no trabalho, pois as competências pertencem ao time ágil, não apenas a algum de seus membros, e por serem complementares entre si;
- Não foi incluído na caracterização da amostra a questão referente aos anos de experiência com desenvolvimento de software, pois não derivaria nenhum dado relevante para o contexto da pesquisa;

- A competência comportamental “Facilidade de comunicação com todos os níveis da organização” foi reformulada para “Facilidade de comunicação com todos” e retirada a menção ao *stakeholder* (CC2);
- A competência comportamental “Flexibilidade e equilíbrio para gerenciar os conflitos do time” teve sua descrição ajustada para estar mais focada no saber lidar com conflitos, ao invés de sempre solucioná-los, o que às vezes pode ser inviável (CC5);
- As competências técnicas relacionadas a alguma prática como, por exemplo, “Dominar a prática de Refatoração de código” e “Dominar a prática de User Stories” tiveram suas descrições refeitas para descrever melhor a prática em questão, e não seu impacto no time ágil;
- A competência comportamental “Responsabilidade pelas atividades e tarefas do time como um todo e por tudo que foi produzido por ele” foi reformulada para “Responsabilidade do time por tudo que foi produzido por ele”, ficando mais focada na prática de propriedade coletiva de código (CC11);
- Foi incluída a competência técnica “Conhecer os valores do desenvolvimento ágil de software” (CT2);
- Foi incluída a competência técnica “Conhecer os princípios do desenvolvimento ágil de software” (CT3);
- A competência técnica referente à integração contínua foi unificada à competência técnica “Dominar práticas e padrões de gerência de configuração de software” (CT12);
- Foram retiradas da caracterização da amostra as perguntas referentes ao grau de formação e às áreas específicas de atuação no setor público;
- Foram incluídas na caracterização da amostra as perguntas referentes ao sexo, participação em time ágil, localização e uso atual de metodologias ágeis.

Em detrimento dessas alterações realizadas, a segunda versão do instrumento de pesquisa foi gerada e está disponível no Apêndice V deste documento.

3.3.4. Disponibilidade do instrumento de pesquisa

Após realizar a etapa de avaliação do questionário e ajustá-lo de acordo com as observações levantadas pelos entrevistados, o questionário foi disponibilizado na internet pela ferramenta Google Drive através do link abaixo:

https://docs.google.com/forms/d/1XTlj6t6TEZ7waOKVP7EJXCQ5TggCgf_B5yRUT6jtmw/viewform?usp=send_form.

O questionário foi divulgado em redes sociais como Facebook e grupos privados relacionados a desenvolvimento ágil de software, por meio de e-mails para instituições, empresas e profissionais ligados ao contexto do trabalho e pessoalmente, realizando contato direito com o respondente.

Sendo disponibilizado para respostas do dia 24/04/2014 até o dia 16/05/2014, o instrumento de pesquisa conseguiu obter 53 respostas em 23 dias de disponibilidade na internet. Através dessa quantidade de respostas, foram coletados os graus de importância para cada competência e foi possível caracterizar a amostra respondente do questionário.

3.3.5. Consistência do instrumento de pesquisa

Com o intuito de verificar a consistência das respostas obtidas pelo questionário, foi utilizado o coeficiente alfa de Cronbach.

O coeficiente alfa foi descrito em 1951 por Lee J. Cronbach e se trata de um índice utilizado para medir a confiabilidade quanto à consistência interna de uma escala ou a magnitude em que os itens de um instrumento estão correlacionados (CORTINA, 1993) (CRONBACH, 1951).

Por se tratar de uma propriedade inerente do padrão de resposta da população estudada, o coeficiente alfa de Cronbach não se restringe apenas às escalas por si só, pois também sofre mudanças segundo a população na qual se aplica a escala (STREINER, 2003).

Para calcular esse coeficiente é necessário aplicar sua fórmula específica ou utilizar uma ferramenta estatística que auxilia nesse procedimento, como foi realizado nesse trabalho. A ferramenta em questão utilizada foi a IBM SPSS

Statistics versão 22 que, além de analisar a confiabilidade do questionário, também fornece uma análise descritiva completa das respostas obtidas pelo questionário.

A partir do coeficiente calculado, é possível determinar a consistência interna e a confiabilidade do questionário, seguindo o raciocínio do Quadro 3. A interpretação do alfa de Cronbach é quase intuitiva, pois os valores variam, em geral, entre zero e 1 e a confiabilidade será maior quanto mais próximo de 1 estiver o coeficiente.

Quadro 3. Confiabilidade do questionário segundo o valor de alfa (Adaptado de (GEORGE e MALLERY, 2003).

Valor de alfa	**Confiabilidade**
Maior do que 0,9	Excelente
Entre 0,8 e 0,9	Bom
Entre 0,7 e 0,8	Aceitável
Entre 0,6 e 0,7	Questionável
Entre 0,5 e 0,6	Pobre
Menor do que 0,5	Inaceitável

Por meio da ferramenta estatística utilizada, o coeficiente alfa de Cronbach encontrado para a pesquisa realizada foi 0,94. Esse cálculo foi baseado nas 53 respostas obtidas no questionário e considerando apenas as 29 perguntas referentes às competências.

Esse resultado colabora muito para a relevância do trabalho, pois, conforme o Quadro 3, o questionário apresenta um excelente nível de confiabilidade, remetendo também a sua consistência interna.

4. APRESENTAÇÃO DOS RESULTADOS

Com o instrumento de pesquisa aplicado e coletados os dados necessários que ele se propôs a buscar, os resultados serão apresentados com base na caracterização da amostra e nos graus de importância obtidos para as competências comportamentais e técnicas.

4.1. CARACTERIZAÇÃO DA AMOSTRA

A caracterização da amostra foi realizada por meio de onze perguntas realizadas ao respondente do questionário e, com isso, foram geradas variáveis que definem e restringem a população amostral do instrumento de pesquisa, sendo elas:

- Sexo;
- Experiência com desenvolvimento ágil de software em meses e/ou anos;
- Atuação e experiências em meses e/ou anos como membro de um time ágil;
- Satisfação quanto à experiência como membro de um time ágil;
- Atuação e experiências em meses e/ou anos como outros papéis, além de ser membro do time ágil;
- Contexto da experiência com desenvolvimento ágil de software;
- Contexto atual de trabalho;
- Uso atual de metodologias ágeis;
- Localização.

Coletadas todas essas variáveis, foi concebido o Quadro 4 com uma síntese dos dados obtidos a cerca dessas variáveis.

Não foi possível restringir no campo texto uma faixa de valores definidos para o tempo das experiências que foram questionadas, por isso algumas respostas foram enviadas sem mencionar a unidade de tempo, apesar de que a obrigatoriedade de se mencionar essa unidade estava explícita na descrição da pergunta. Com isso, esses dados não foram desconsiderados, sendo agrupados como “Dados sem unidade” no Quadro 4.

Quadro 4. Caracterização da amostra (O autor).

Variável	**Alternativas**	**Resultados**	
		Absoluto	**Percentual**
Sexo	Masculino	44	83%
	Feminino	9	17%
Experiência com desenvolvimento ágil de software	Nenhuma	4	7,5%
	1 a 6 meses	8	15%
	7 a 12 meses	4	7,5%
	1 a 3 anos	20	38%
	4 a 5 anos	9	17%
	Mais de 5 anos	3	6%
	Dados sem unidade	5	9%
Atuação como membro do time ágil	Sim	43	81%
	Não	10	19%
Experiência como membro do time ágil	Nenhuma	10	19%
	1 a 6 meses	18	34%
	7 a 12 meses	2	4%
	1 a 3 anos	13	24,5%
	Mais de 3 anos	4	7,5%
	Dados sem unidade	6	11%

Variável	Alternativas	Resultados	
		Absoluto	Percentual
Satisfação com a experiência como membro do time ágil	Sim	42	79%
	Não	1	2%
	Não se aplica	10	19%
Atuação como outros papéis	*Scrum Master*	25	30%
	Product Owner	13	16%
	Stakeholder	8	10%
	Patrocinador	3	4%
	Estudante	33	40%
Experiência com outros papéis desempenhados	Nenhuma	4	7,5%
	1 a 6 meses	17	32,5%
	7 a 12 meses	1	2%
	1 a 3 anos	20	38%
	Mais de 3 anos	6	11%
	Dados sem unidade	5	9%
Contexto da experiência com desenvolvimento ágil de software	Setor público	19	28%
	Iniciativa privada	25	36%
	Área Acadêmica	25	36%
Contexto atual de desenvolvimento de	Setor público	22	41,5%
	Iniciativa privada	20	37,5%

Variável	Alternativas	Resultados	
		Absoluto	Percentual
software	Área Acadêmica	11	21%
Uso de metodologias ágeis atualmente	Sim	38	72%
	Não	15	28%
Localização	DF	41	77%
	SP	3	6%
	PI	5	9%
	Outros (PA, MA, GO e CE)	4	8%

Na variável “Localização” do Quadro 4, foram agrupados a quantidade de respondentes dos estados do Ceará, Goiás, Maranhão e Pará, pois para cada estado houve apenas um respondente.

A partir dessa síntese dos dados obtidos para caracterizar os respondentes da pesquisa, percebe-se como pontos relevantes que a maioria dos participantes (81%) já foram membros de times ágeis, sendo boa parte deles (98%) satisfeita com essa experiência, 41,5% de todos os respondentes são do setor público, o estado que mais contribui com respostas para a pesquisa (77%) foi o Distrito Federal e a maioria dos participantes (72%) continua utilizando metodologias ágeis atualmente.

Outros pontos também chamaram atenção como, por exemplo, a pouca experiência com desenvolvimento ágil de software, onde 38% dos respondentes tem experiência de 1 a 3 anos. A experiência como membro do time ágil também foi baixa, pois 34% dos respondentes tem experiência de 1 a 6 meses.

Quanto ao desempenho de outros papéis além do time ágil, 40% foram apenas estudantes, seguido de 30% que afirmaram ter exercido o papel de *Scrum Master*. A experiência com esses outros papéis também foi baixa, pois 38%

afirmaram ter experiência de 1 a 3 anos, seguido de 32,5% que tiveram experiência de 1 a 6 meses.

Quanto aos contextos no qual foram aplicadas essas experiências com desenvolvimento ágil de software, destacou-se a iniciativa privada com 36% dos respondentes e a área acadêmica com a mesma porcentagem.

Mesmo diante desses resultados, foram encontradas inconsistências nas informações fornecidas nesta pesquisa. Na satisfação com a experiência como membro de um time ágil, houve uma resposta em que o respondente afirmou não ter sido membro do time e, mesmo assim, opinou não ser satisfatória sua experiência como membro de um time ágil, por isso essa resposta foi corrigida e colocada como “Não se aplica”.

Na localização, um respondente informou sua cidade, ao invés da Unidade da Federação (UF), por isso essa resposta foi corrigida, sendo alterada para a unidade federativa da cidade em questão.

Na experiência como membro do time ágil, alguns participantes que não foram membros de um time ágil apenas repetiram sua experiência com desenvolvimento ágil de software nesse campo, por isso essas respostas foram corrigidas e alteradas para o valor 0, que corresponde à opção “Não se aplica”.

4.2. COMPETÊNCIAS COMPORTAMENTAIS

A parte do questionário destinada às competências comportamentais é disposta de 13 assertivas ou competências para serem avaliadas conforme a opinião do respondente e, por isso, serão apresentadas uma por uma.

Cada pergunta contém um enunciado com o nome da competência comportamental a ser julgada e uma descrição dela para guiar o respondente, caso o enunciado não seja suficiente para entender o propósito daquela determinada competência.

Os dados detalhados e gráficos gerados para as respostas de cada competência comportamental estão disponíveis no Apêndice VI deste documento.

4.2.1. Capacidade de compartilhar e transferir o conhecimento obtido

Como resultado da avaliação dessa competência, que está relacionada ao compartilhamento de experiências e transferência do que foi aprendido entre o time ágil, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 68% dos respondentes.

4.2.2. Facilidade de comunicação com todos

Como resultado da avaliação dessa competência, que está relacionada à interação e articulação com todos os comprometidos com o projeto, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 62% dos respondentes.

4.2.3. Capacidade de trabalhar em equipe e/ou junto com outro membro da equipe

Como resultado da avaliação dessa competência, que está relacionada à facilidade de se adequar ao trabalho em grupo e à prática de programação em pares, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 58% dos respondentes.

4.2.4. Disposição para ajudar o time ou membros do time sempre que for necessário

Como resultado da avaliação dessa competência, que está relacionada à participação ativa e colaborativa de todos os membros do time ágil, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 49% dos respondentes.

4.2.5. Flexibilidade e equilíbrio para gerenciar os conflitos do time

Como resultado da avaliação dessa competência, que está relacionada ao saber lidar com conflitos para que sejam melhor solucionados ou contornados, foi

possível notar que as respostas tenderam para “Muito importante”, concentrando 43% dos respondentes.

4.2.6. Capacidade de cumprir com seus compromissos

Como resultado da avaliação dessa competência, que está relacionada aos compromissos feitos pelo time quanto aos objetivos, às entregas e aos padrões de qualidade do projeto, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 64% dos respondentes.

4.2.7. Proatividade e iniciativa para a busca de conhecimento e designação de tarefas

Como resultado da avaliação dessa competência, que está relacionada ao interesse e responsabilidade por atividades que possui afinidade e procura de conhecimento, foi possível notar que as respostas tenderam para “Muito importante”, concentrando 45% dos respondentes.

4.2.8. Capacidade de estabelecer relacionamentos de confiança com os membros do time

Como resultado da avaliação dessa competência, que está relacionada à criação de relacionamentos transparentes e maduros, pautados na sinceridade e responsabilidade, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 47% dos respondentes.

4.2.9. Capacidade de tomar decisões diante de situações de incerteza

Como resultado da avaliação dessa competência, que está relacionada à autonomia suficiente para determinar, a nível técnico, a maneira que o projeto será implementado, foi possível notar que as respostas tenderam para “Muito importante”, concentrando 49% dos respondentes.

4.2.10. Visão focada nos objetivos do time e compartilhada com os membros dele

Como resultado da avaliação dessa competência, que está relacionada ao modelo mental compartilhado entre todos os membros do time para evitar trabalhos extras e desnecessários, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 45% dos respondentes.

4.2.11. Responsabilidade do time por tudo que foi produzido por ele

Como resultado da avaliação dessa competência, que está relacionada à utilização da prática de propriedade coletiva, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 30% dos respondentes.

4.2.12. Capacidade de evolução e aprendizado com o progresso do trabalho

Como resultado da avaliação dessa competência, que está relacionada à melhoria contínua, mitigação do gargalos dentro do processo e excelência técnica, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 60% dos respondentes.

4.2.13. Capacidade de auto-gerenciar suas atividades e trabalho

Como resultado da avaliação dessa competência, que está relacionada à organização e gerenciamento do trabalho feito pelo time ágil sem necessidade de um gestor para controlar, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 57% dos respondentes.

4.3. COMPETÊNCIAS TÉCNICAS

A parte do questionário destinada às competências técnicas é disposta 16 assertivas ou competências para serem avaliadas conforme a opinião do respondente e, por isso, serão apresentadas uma por uma.

Assim como foi feito com as competências comportamentais, cada pergunta contem um enunciado com o nome da competência técnica a ser julgada e uma

descrição dela para guiar o respondente, caso o enunciado não seja suficiente para entender o propósito daquela determinada competência.

Os dados detalhados e gráficos gerados para as respostas de cada competência comportamental está disponível no Apêndice VI deste documento.

4.3.1. Conhecer a instituição em que trabalha para desempenhar seu papel

Como resultado da avaliação dessa competência, que está relacionada ao conhecimento das práticas e normas utilizadas pelo time e as pessoas envolvidas no projeto, foi possível notar que as respostas tenderam para “Mediamente importante”, concentrando 40% dos respondentes.

4.3.2. Conhecer os valores do desenvolvimento ágil de software

Como resultado da avaliação dessa competência, que está relacionada ao conhecimento dos valores difundidos pelo Manifesto ágil, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 45% dos respondentes.

4.3.3. Conhecer os princípios do desenvolvimento ágil de software

Como resultado da avaliação dessa competência, que está relacionada ao conhecimento dos princípios difundidos pelo Manifesto ágil, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 51% dos respondentes.

4.3.4. Conhecer metodologias ágeis de desenvolvimento de software

Como resultado da avaliação dessa competência, que está relacionada ao conhecimento do processo, cerimônias, regras e artefatos que o time ágil segue e utiliza, foi possível notar que as respostas tenderam para “Muito importante” e “Extremamente importante”, concentrando 36% dos respondentes em cada alternativa.

4.3.5. Dominar linguagens de programação

Como resultado da avaliação dessa competência, que está relacionada ao conhecimento e habilidade com linguagens de programação, como, por exemplo, Java e C/C++, foi possível notar que as respostas tenderam para “Mediamente importante”, concentrando 42% dos respondentes.

4.3.6. Dominar uso de padrões de codificação

Como resultado da avaliação dessa competência, que está relacionada à capacidade de escrever o código utilizando padrões de escrita e identação, visando facilitar a interpretação e manutenção por outro membro do time, foi possível notar que as respostas tenderam para “Muito importante”, concentrando 38% dos respondentes.

4.3.7. Dominar a prática de TDD e seus padrões

Como resultado da avaliação dessa competência, que está relacionada à capacidade de escrever testes antes do código e cumpri-los com o mínimo de código possível, foi possível notar que as respostas tenderam para “Muito importante”, concentrando 40% dos respondentes.

4.3.8. Dominar a prática de refatoração de código

Como resultado da avaliação dessa competência, que está relacionada à capacidade de modificar um sistema de software de modo que não altere o comportamento externamente observável e ainda melhore sua estrutura interna, foi possível notar que as respostas tenderam para “Muito importante”, concentrando 49% dos respondentes.

4.3.9. Dominar a prática de *User stories*

Como resultado da avaliação dessa competência, que está relacionada à capacidade de descrever funcionalidades valiosas para os usuário de um sistema, sendo compostas por um breve descrição, conversas ou debates e critérios de

aceitação, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 38% dos respondentes.

4.3.10. Dominar práticas de desenvolvimento da arquitetura do sistema

Como resultado da avaliação dessa competência, que está relacionada à capacidade de definir o framework ou estrutura do sistema, seu comportamento, suas propriedades e relacionamentos com outros sistemas, foi possível notar que as respostas tenderam para “Muito importante”, concentrando 42% dos respondentes.

4.3.11. Dominar práticas de desenvolvimento do banco de dados do sistema

Como resultado da avaliação dessa competência, que está relacionada à capacidade de criar e modelar um banco de dados SQL, por exemplo, foi possível notar que as respostas tenderam para “Mediamente importante”, concentrando 43% dos respondentes.

4.3.12. Dominar práticas e padrões de gerência de configuração de software

Como resultado da avaliação dessa competência, que está relacionada à capacidade de manter a integridade da configuração através de práticas como integração de contínua e controle de versão, foi possível notar que as respostas tenderam para “Muito importante”, concentrando 42% dos respondentes.

4.3.13. Capacidade de construir e configurar o ambiente de desenvolvimento

Como resultado da avaliação dessa competência, que está relacionada à capacidade de construir e manter a infraestrutura necessária para comportar o desenvolvimento do projeto, foi possível notar que as respostas tenderam para “Muito importante”, concentrando 34% dos respondentes.

4.3.14. Dominar a prática de reuso de software

Como resultado da avaliação dessa competência, que está relacionada à capacidade de aproveitar componentes de sistemas finalizados para a construção de um outro novo e diferente, foi possível notar que as respostas tenderam para “Muito importante”, concentrando 32% dos respondentes.

4.3.15. Capacidade de negociar o escopo do trabalho

Como resultado da avaliação dessa competência, que está relacionada à capacidade de deliberar e ponderar sobre o escopo do projeto que será trabalhado para não sobrecarregar o time ágil, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 43% dos respondentes.

4.3.16. Capacidade de estimar esforço de acordo com suas limitações

Como resultado da avaliação dessa competência, que está relacionada à capacidade de estimar o esforço para realizar as users stories adequadamente considerando as limitações técnicas e físicas do time ágil, foi possível notar que as respostas tenderam para “Extremamente importante”, concentrando 45% dos respondentes.

4.4. CONSIDERAÇÕES DOS RESPONDENTES

Ao término da avaliação da importância das competências, foi disponibilizado na parte “Observações” do questionário um campo para que o respondente colocasse suas sugestões de competências que não foram tratadas, críticas que poderiam melhorar o trabalho ou elogios.

Um respondente ressaltou que deveria haver alguma competência voltada à visão, missão e aos objetivos da empresa em que o time ágil atua. Outro respondente sugeriu que essa mesma metodologia de pesquisa fosse aplicada para um estudo de outras metodologias de desenvolvimento de software, inclusive as que não são ágeis.

Quanto aos elogios, um respondente afirmou que não tinha nenhuma sugestão e que o instrumento de pesquisa estava completo. Já outro participante ressaltou que as competências estão em conformidade com o desenvolvimento ágil e que, através dessas competências, percebe que o trabalho em equipe do time ágil não é ser apenas um programador completo, mas sim uma equipe constituída de membros que se completam e que unidos geram resultados efetivos e eficazes.

4.5. ANÁLISE DOS RESULTADOS

No contexto geral as competências avaliadas tiveram um grau de importância tendendo para as escalas mais a direita, que são mediamente, muito e extremamente importante. Isso indica que as competências demonstraram ser consistentes e estarem em concordância com o cenário do desenvolvimento ágil de software, cabendo ressaltar que nenhuma delas foi rechaçada ou questionada pelas pessoas que responderam o questionário.

Mesmo diante desses resultados positivos, foi utilizado um critério para analisar tanto o questionário quanto os graus de importância obtidos em cada competência.

4.5.1. Critério de análise dos dados

Para analisar a pesquisa realizada, foi utilizado como critério a curva da distribuição normal, melhor ilustrada na Figura 4.

Com resultados obtidos por meio da aplicação do instrumento de pesquisa e para melhor interpretar o que foi encontrado para cada item do questionário, foram plotados histogramas, que são representações gráficas formada por conjuntos de retângulos justapostos, cujas bases se localizam sobre o eixo horizontal, de maneira que seus pontos médios coincidam com os pontos médios dos intervalos de classe (CRESPO, 2002). A largura de cada retângulo é igual à amplitude de cada intervalo de classe, representada pelo eixo horizontal, que corresponde aos graus de importância das competências no questionário; já a altura de cada retângulo, deve ser proporcional à frequência da classe representada, correspondendo à quantidade de respostas obtida (CRESPO, 2002).

A partir dos histogramas gerados, foi possível verificar a curva normal de cada item do questionário, sendo essa curva a representação gráfica da distribuição normal (WALPOLE, 2009).

Conforme os gráficos feitos, foram selecionados para análise as competências que apresentaram curva semelhante à curva da distribuição normal representada pela Figura 4.

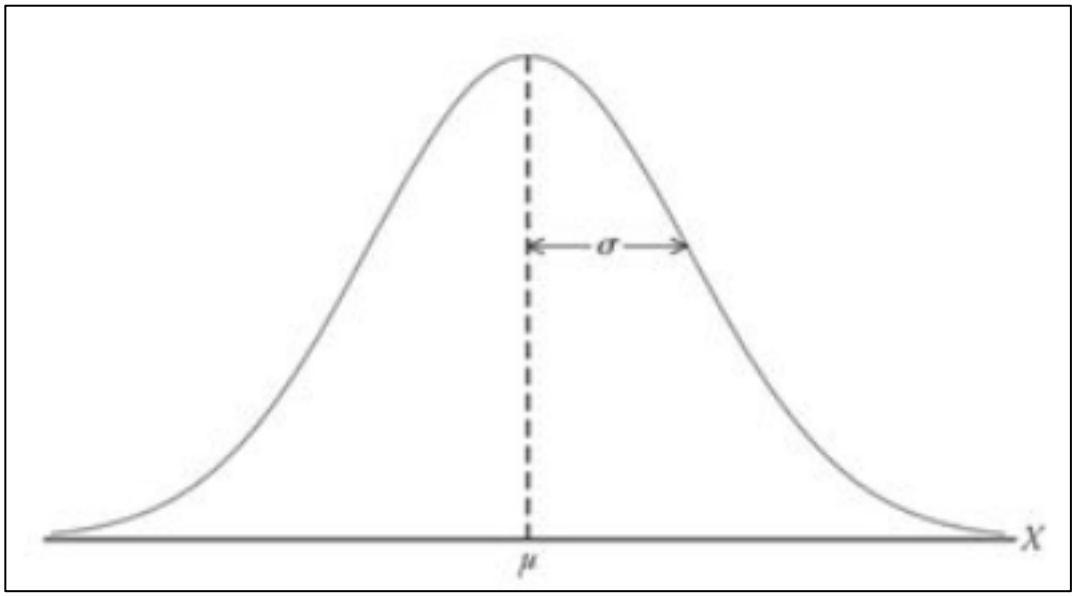


Figura 4. Curva da distribuição normal (WALPOLE, 2009).

Como nenhuma competência foi considerada como “Nada importante” ou “Pouco importante” e a quantidade de respostas para essas escalas foi minoritária para todas as competências, as competências que tiveram o ápice da curva normal na escala do meio (“Mediamente importante”) foram selecionadas para uma análise isolada.

Sendo assim, o conjunto de competências que foram selecionadas para análise é formado pelos itens 1, 5 e 11 disposto no questionário na parte das competências técnicas.

4.5.2. Análise dos dados

Após a aplicação dos critérios, foram selecionados três itens do questionário para uma análise mais profunda e isolada. Essa análise busca, além de entender o

comportamento da competência selecionada e sua motivação através de seu histograma, derivar informações adicionais a cerca desse determinado comportamento.

Mesmo sem apresentar a curva normal em seus gráficos, foram adicionados mais dois itens do questionário para serem analisados detalhadamente, sendo eles os itens 9 e 10 do questionário na parte das competências técnicas. Por conta do comportamento observado em seus histogramas, achou-se interessante debatê-los para, além de gerar discussões construtivas para o contexto do trabalho, entender esse comportamento e sua motivação.

A análise de cada item ou competência selecionada do questionário foi baseada na caracterização da amostra que, a partir de quadros e diagramas, foram filtradas e utilizadas as seguintes características para detalhar o resultado obtido:

- Quantidades de respondentes absoluta (Ab) e percentual (Per) para os intervalos de escala;
- Quantidade absoluta dessa parcela de respondentes dividida pelos contextos onde foi aplicada essa experiência com desenvolvimento ágil, sendo eles:
    - Iniciativa privada;
    - Setor público;
    - Área acadêmica;
- Quantidade absoluta (Ab) e percentual (Per) de respondentes que já foram membros de um time ágil.

Os quantitativos dispostos acima referentes aos valores absoluto e percentual de respostas para as escalas e dos respondentes que já foram membros de um time ágil foram expressos em quadros, enquanto os valores relacionados aos contextos de aplicação das experiências foram ilustrados em gráficos, mais especificamente em diagramas de *Venn*.

Salvo as exceções, foram selecionadas para análise apenas as competências com o ápice da curva normal na escala “Mediamente importante” e para identificar sua importância observou-se os extremos de seus gráficos, sendo as escalas mais à esquerda (“Nada importante” e “Pouco importante”) e à direita (“Muito importante” e “Extremamente importante”).

4.5.2.1. Conhecer a instituição em que trabalha para desempenhar seu papel

O item selecionado, que representa a Competência Técnica 1 (CT1), possui sua distribuição de respostas obtidas ilustrada no Gráfico 3 e a análise feita com base na caracterização dos respondentes disposta no Quadro 5 e Figuras 5 e 6.

Gráfico 3. Conhecer a instituição em que trabalha para desempenhar seu papel (O autor).

Diante dos dados constantes no Quadro 5, é possível visualizar que os respondentes que consideram a competência como “Nada importante” e “Pouco importante” correspondem a 6% do total, sendo três participantes.

Quadro 5. Análise da CT1 (O autor).

Grau de Importância	Quantidade de respondentes		Membros do time ágil	
	Abs	Per	Abs	Per
1 e 2	3	6%	3	100%
4 e 5	29	54%	22	76%

Com base no Figura 5, é possível verificar que os perfis preponderantes dos respondentes que atribuíram a CT1 como “Nada importante” e “Pouco importante” , que foram destacados de vermelho, são de profissionais que utilizaram desenvolvimento ágil de software no setor público e/ou na iniciativa privada, pois um deles afirmou ter utilizado apenas na iniciativa privada, outro utilizou na iniciativa privada e no setor público e o último no setor público e na área acadêmica.

Já os respondentes que consideram a competência como “Muito importante” e “Extremamente Importante” correspondem a 54% do total, sendo 29 participantes.

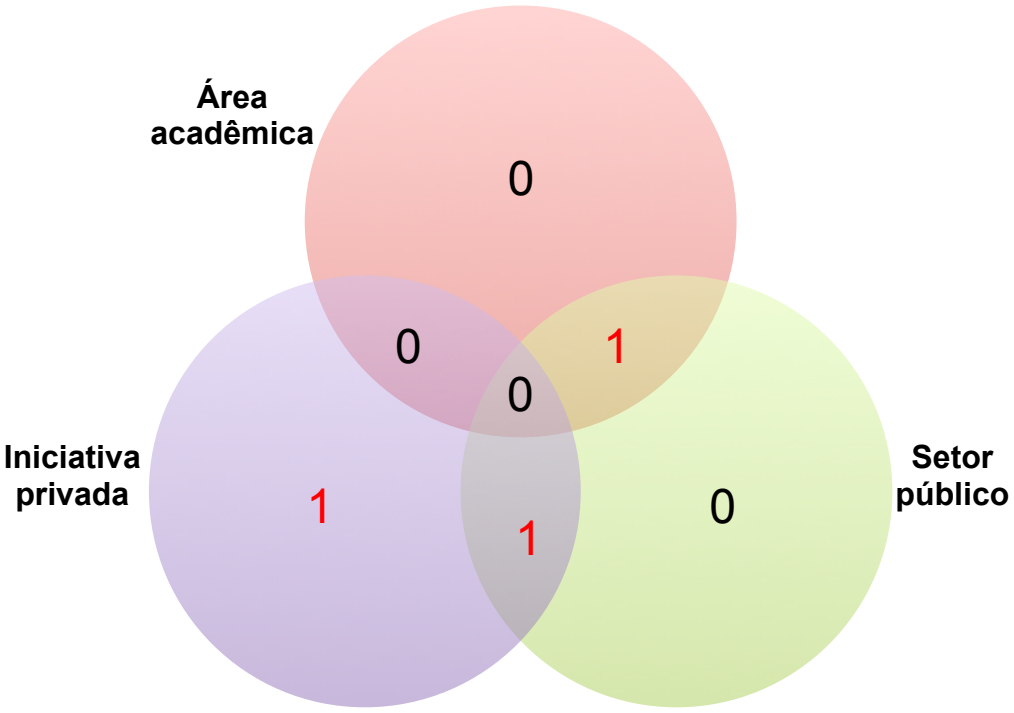


Figura 5. Quantidade de respondentes por contexto para os graus de importância 1 e 2 da CT1 (O autor).

Considerando o Figura 6, é possível verificar que os perfis preponderantes dos respondentes que atribuíram a CT1 como “Muito importante” e “Extremamente importante”, que foram destacados de vermelho, são de profissionais que utilizaram desenvolvimento ágil de software na iniciativa privada ou no setor público ou na área acadêmica, pois oito deles afirmaram ter utilizado apenas na iniciativa privada, outros oito utilizaram apenas no setor público e mais oito deles disseram ter utilizado apenas na área acadêmica.

É válido ressaltar também que, daqueles que consideraram a competência como “Nada importante” e “Pouco importante”, todos os três já foram membros de times ágeis; já os que consideraram a competência como “Muito importante” e “Extremamente Importante”, 22 deles já participaram de times ágeis e fazem parte da maioria que classifica como relevante a CT1.

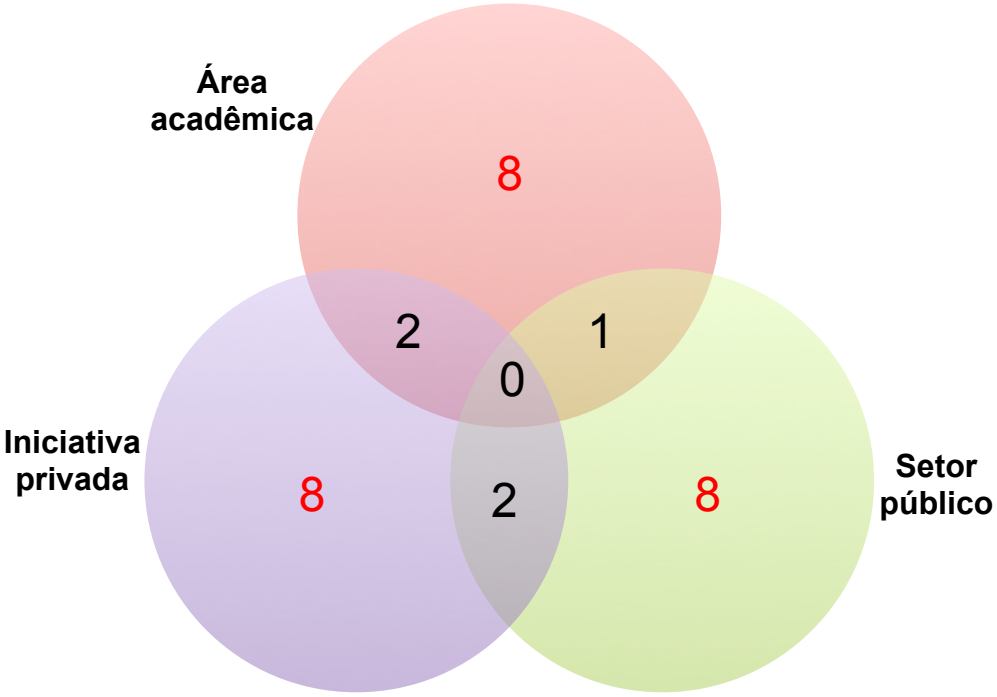


Figura 6. Quantidade de respondentes por contexto para graus de importância 4 e 5 da CT1 (O autor).

A partir dos gráficos, quadros, figuras e interpretações feitas, nota-se que a competência “Conhecer a instituição em que trabalha para desempenhar seu papel” (CT1), apesar de apresentar um pico de frequência de respostas na escala “Mediamente importante”, possui a maior parte das respostas nas escalas à direita que expressam a importância da competência, sendo que a maioria dessas respostas vem de membros de times ágeis, o que torna essa competência importante para o contexto do desenvolvimento ágil de software e consistente diante da pesquisa realizada.

4.5.2.2. Dominar linguagens de programação

O item selecionado, que representa a Competência Técnica 5 (CT5), possui sua distribuição de respostas obtidas ilustrada no Gráfico 4 e a análise feita com base na caracterização dos respondentes disposta no Quadro 6 e Figuras 7 e 8.

Gráfico 4. Dominar linguagens de programação (O autor).

Diante dos dados constantes no Quadro 6, é possível visualizar que os respondentes que consideram a competência como “Nada importante” e “Pouco importante” correspondem a 4% do total, sendo dois participantes.

Quadro 6. Análise da CT5 (O autor).

Grau de Importância	**Quantidade de respondentes**		**Membros do time ágil**	
	Abs	**Per**	**Abs**	**Per**
1 e 2	2	4%	2	100%
4 e 5	29	54%	21	72%

Com base na Figura 7, é possível verificar que o perfil preponderante dos respondentes que atribuíram a CT5 como “Nada importante” e “Pouco importante”, que foi destacado de vermelho, é de um profissional que utilizou desenvolvimento ágil de software na área acadêmica, pois um deles afirmou ter utilizado apenas na área acadêmica e o outro utilizou na área acadêmica e na iniciativa privada.

Já os respondentes que consideram a competência como “Muito importante” e “Extremamente Importante” correspondem a 54% do total, sendo 29 participantes.

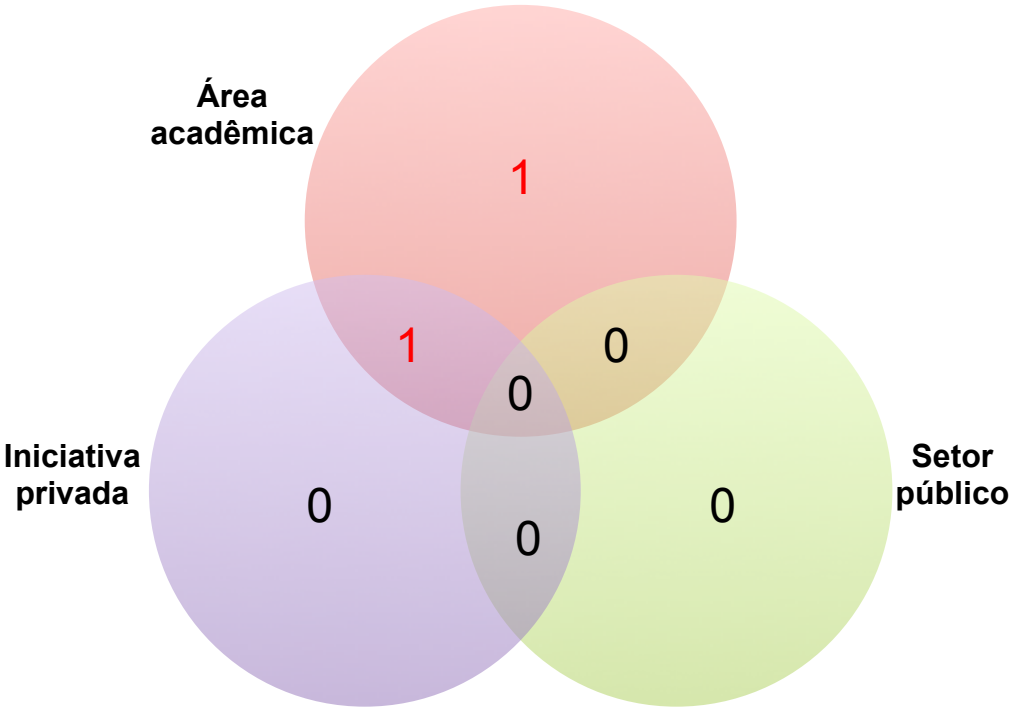


Figura 7. Quantidade de respondentes por contexto para os graus de importância 1 e 2 da CT5 (O autor).

Considerando a Figura 8, é possível verificar que o perfil preponderante dos respondentes que atribuíram a CT5 como “Muito importante” e “Extremamente importante”, que foi destacado de vermelho, é de um profissional que utilizou desenvolvimento ágil de software na área acadêmica, pois oito deles afirmaram ter utilizado apenas na área acadêmica.

É válido ressaltar também que, daqueles que consideraram a competência como “Nada importante” e “Pouco importante”, todos os dois já foram membros de times ágeis; já os que consideraram a competência como “Muito importante” e “Extremamente Importante”, 21 deles já participaram de times ágeis e fazem parte da maioria que classifica como relevante a CT5.

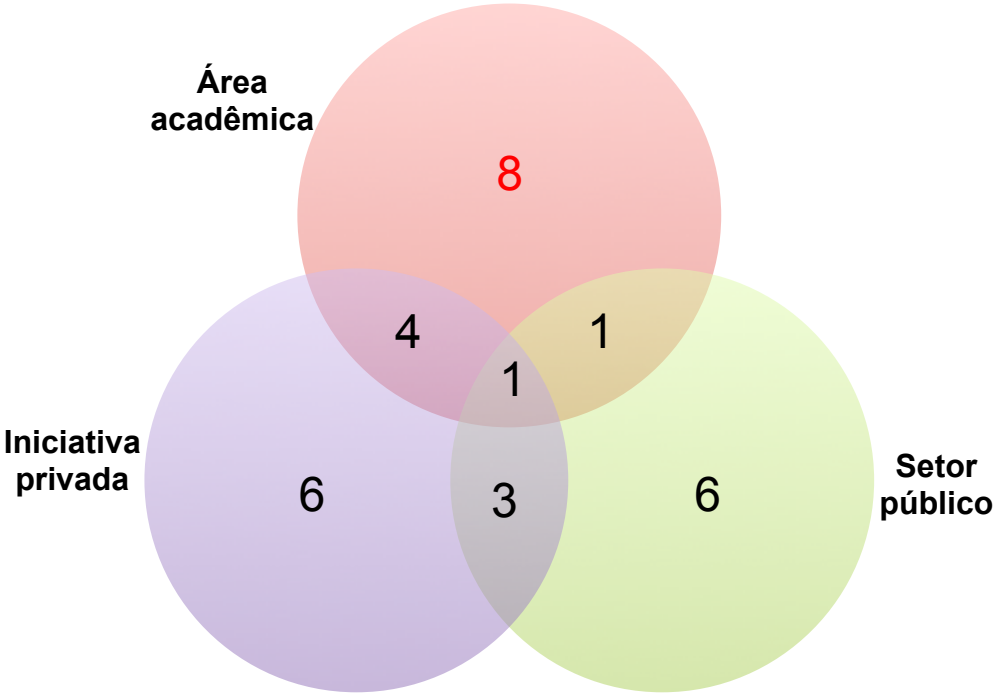
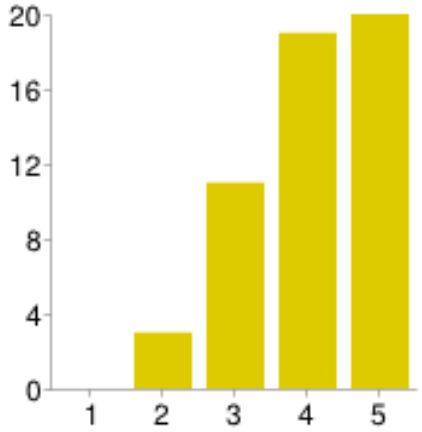


Figura 8. Quantidade de respondentes por contexto para graus de importância 4 e 5 da CT5 (O autor).

A partir dos gráficos, quadros e interpretações feitos, nota-se que a competência “Dominar linguagens de programação” (CT5), apesar de apresentar um pico de frequência de respostas na escala “Mediamente importante”, possui a maior parte das respostas nas escalas à direita que expressam a importância da competência, sendo que a maioria dessas respostas vem de membros de times ágeis, o que torna essa competência importante para o contexto do desenvolvimento ágil de software e consistente diante da pesquisa realizada.

4.5.2.3. Dominar a prática de *User stories*

O item selecionado, que representa a Competência Técnica 9 (CT9), possui sua distribuição de respostas obtidas ilustrada no Gráfico 5 e a análise feita com base na caracterização dos respondentes disposta no Quadro 7 e Figuras 9 e 10.



1	**0**	0%
2	**3**	6%
3	**11**	21%
4	**19**	36%
5	**20**	38%

Gráfico 5. Dominar a prática de User stories (O autor).

Diante dos dados constantes no Quadro 7, é possível visualizar que os respondentes que consideram a competência como “Nada importante” e “Pouco importante” correspondem a 4% do total, sendo dois participantes.

Quadro 7. Análise da CT9 (O autor).

Grau de Importância	**Quantidade de respondentes**		**Membros do time ágil**	
	Abs	**Per**	**Abs**	**Per**
1 e 2	3	6%	3	100%
4 e 5	39	74%	31	79%

Com base na Figura 9, é possível verificar que o perfil preponderante dos respondentes que atribuíram a CT9 como “Nada importante” e “Pouco importante”, que foi destacado de vermelho, é de um profissional que utilizou desenvolvimento ágil de software no setor público, pois um deles afirmou ter utilizado apenas no setor público e o outro utilizou no setor público e na área acadêmica.

Já os respondentes que consideram a competência como “Muito importante” e “Extremamente Importante” correspondem a 74% do total, sendo 39 participantes.

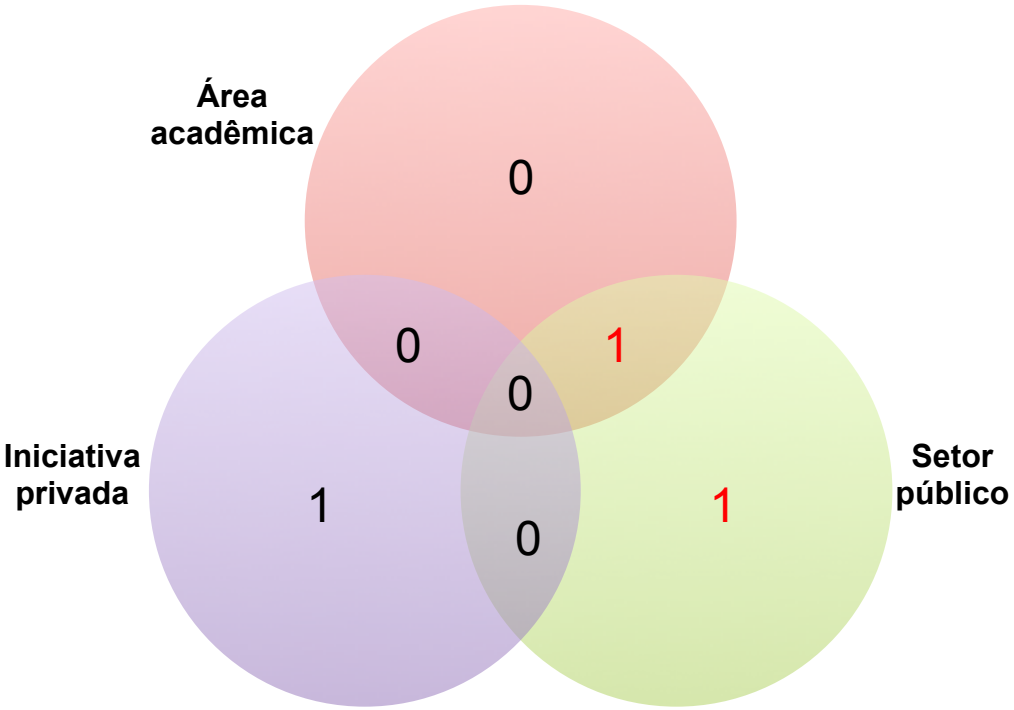


Figura 9. Quantidade de respondentes por contexto para os graus de importância 1 e 2 da CT9 (O autor).

Considerando o Figura 10, é possível verificar que o perfil preponderante dos respondentes que atribuíram a CT9 como “Muito importante” e “Extremamente importante”, que foi destacado de vermelho, é de um profissional que utilizou desenvolvimento ágil de software na área acadêmica, pois oito deles afirmaram ter utilizado apenas na área acadêmica.

É válido ressaltar também que, daqueles que consideraram a competência como “Nada importante” e “Pouco importante”, todos os dois já foram membros de times ágeis; já os que consideraram a competência como “Muito importante” e “Extremamente Importante”, 21 deles já participaram de times ágeis e fazem parte da maioria que classifica como relevante a CT5.

Conforme exposto no referencial teórico deste trabalho, no que tange às *users stories* o time ágil é responsável por selecionar quais delas serão implementadas, como serão e atualizá-las constantemente, sendo a criação desses requisitos responsabilidade do *Product* Owner. Mesmo assim, é questionável se na realidade a responsabilidade de criação das *users stories* seja realizada pelo time

ágil, visto que diante das respostas obtidas essa competência evidenciou uma expressiva importância para um time ágil.

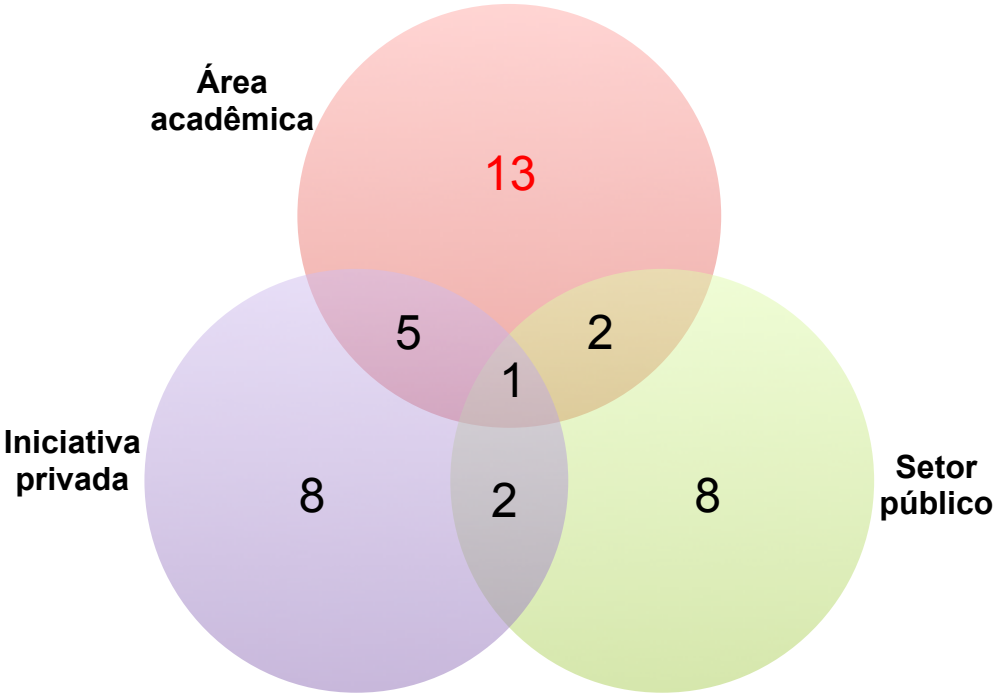


Figura 10. Quantidade de respondentes por contexto para graus de importância 4 e 5 da CT9 (O autor).

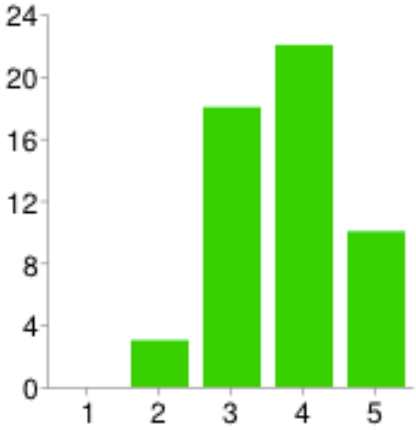
Segundo Cohn (2004), um projeto ideal teria uma única pessoa para priorizar o trabalho para o time ágil, de maneira onisciente responder suas questões e escrever todas as histórias, porém isso é esperar muito de uma pessoa apenas, sendo essas as responsabilidades de um *Product Owner*, o que o torna um papel diferenciado.

Com isso, é passível a um time ágil pequeno ou com recurso limitado de não ter um *Product Owner* para auxiliá-lo, cabendo ao time ter que desempenhar também as responsabilidades desse papel e configurando um tópico que poderá ser melhor discutido em pesquisas futuras.

4.5.2.4. Dominar práticas de desenvolvimento da arquitetura do sistema

O item selecionado, que representa a Competência Técnica 10 (CT10), possui sua distribuição de respostas obtidas ilustrada no Gráfico 6 e a análise feita

com base na caracterização dos respondentes disposta no Quadro 8 e Figuras 11 e 12.



1	**0**	0%
2	**3**	6%
3	**18**	34%
4	**22**	42%
5	**10**	19%

Gráfico 6. Dominar práticas de desenvolvimento da arquitetura do sistema (O autor).

Diante dos dados constantes no Quadro 8, é possível visualizar que os respondentes que consideram a competência como “Nada importante” e “Pouco importante” correspondem a 6% do total, sendo três participantes.

Quadro 8. Análise da CT10 (O autor).

Grau de Importância	**Quantidade de respondentes**		**Membros do time ágil**	
	Abs	**Per**	**Abs**	**Per**
1 e 2	3	6%	3	100%
4 e 5	32	60%	24	75%

Com base na Figura 11, é possível verificar que o perfil preponderante dos respondentes que atribuíram a CT10 como “Nada importante” e “Pouco importante”, que foi destacado de vermelho, é de um profissional que utilizou desenvolvimento ágil de software na iniciativa privada, pois dois deles afirmaram ter utilizado apenas na iniciativa privada e o outro utilizou apenas no setor público.

Já os respondentes que consideram a competência como “Muito importante” e “Extremamente Importante” correspondem a 60% do total, sendo 32 participantes.

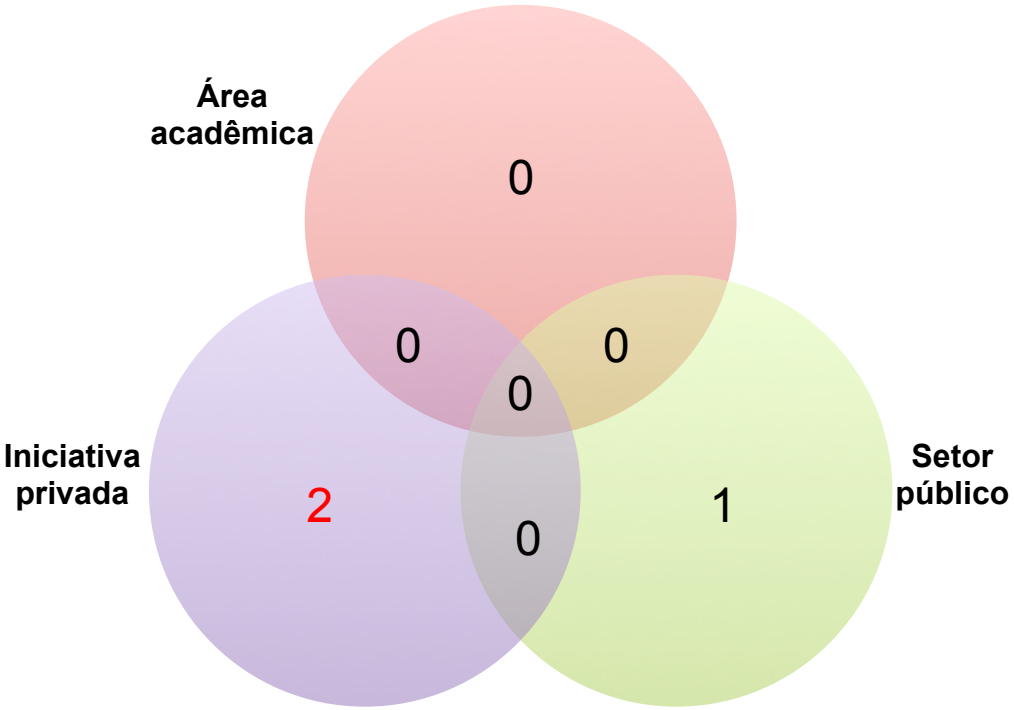


Figura 11. Quantidade de respondentes por contexto para os graus de importância 1 e 2 da CT10 (O autor).

Considerando a Figura 12, é possível verificar que o perfil preponderante dos respondentes que atribuíram a CT10 como “Muito importante” e “Extremamente importante”, que foi destacado de vermelho, é de um profissional que utilizou desenvolvimento ágil de software na área acadêmica, pois dez deles afirmaram ter utilizado apenas na área acadêmica.

É válido ressaltar também que, daqueles que consideraram a competência como “Nada importante” e “Pouco importante”, todos os três já foram membros de times ágeis; já os que consideraram a competência como “Muito importante” e “Extremamente Importante”, 24 deles já participaram de times ágeis e fazem parte da maioria que classifica como relevante a CT10.

Conforme exposto no referencial teórico, no que tange à arquitetura o time ágil juntamente com o pessoal de negócios elabora a metáfora ou uma história abrangente e simplória que descreve o que o sistema deverá fazer e ajuda a todos os envolvidos a compreender melhor os elementos básicos do sistema e seus relacionamentos (BECK, 2004). A partir dessa metáfora, a arquitetura deve emergir naturalmente com a implementação das users stories mais prioritárias, fazendo

desnecessário a criação de um documento para retratar toda a arquitetura do sistema ou modelá-la (AMBLER, 2002).

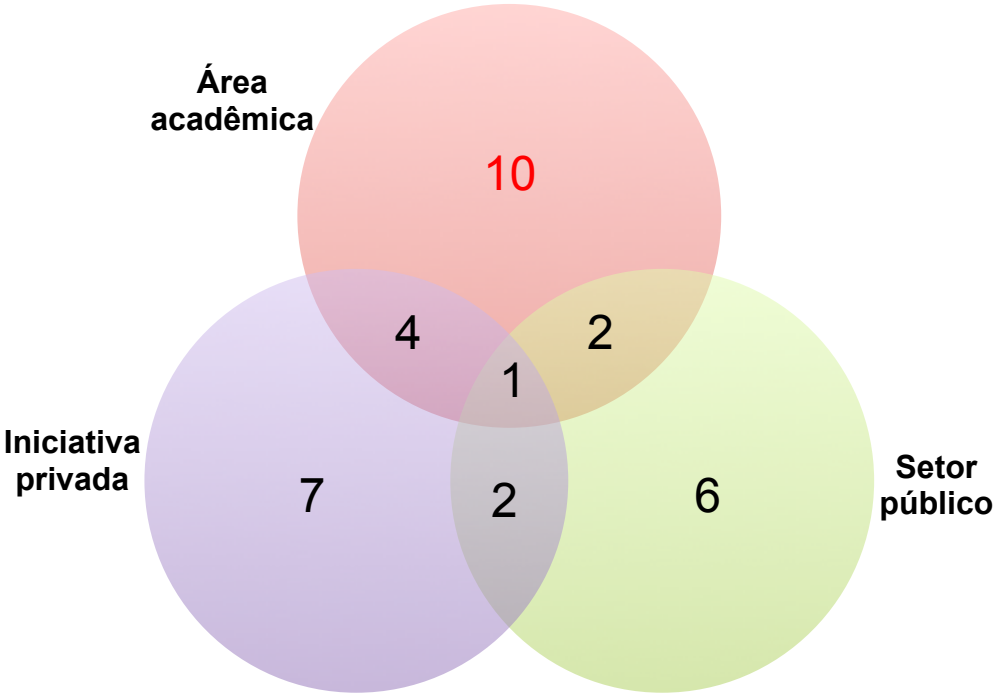


Figura 12. Quantidade de respondentes por contexto para graus de importância 4 e 5 da CT10 (O autor).

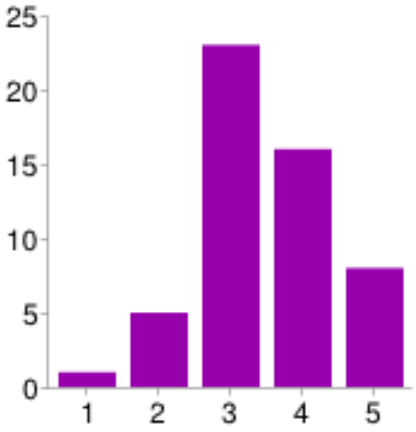
Mesmo assim, é questionável se na realidade os times ágeis modelam ou documentam a arquitetura antes, depois ou durante a codificação, visto que diante das resposta obtidas essa competência evidenciou uma expressiva importância para um time ágil.

Um dos entrevistados que participou da validação do instrumento de pesquisa, fez uma observação nesse aspecto, dizendo ser um tópico discutível já que depende muito do contexto e a necessidade do projeto que está sendo desenvolvido. Ele afirmou ter trabalhado em um projeto como membro do time ágil, onde o time fez uso da metáfora e não realizou qualquer modelagem ou documentação da arquitetura, seguindo exatamente como as metodologias ágeis pregam, porém, na mesma empresa, ouviu relatos de colegas de trabalho que trabalharam em um projeto, onde o time ágil teve a necessidade de modelar e documentar toda a arquitetura do sistema.

Com isso, é passível a um time ágil ter ou não necessidade de modelar e documentar a arquitetura do sistema que está sendo desenvolvido, verificando se isso é importante para o projeto e configurando um tópico que poderá ser melhor discutido em pesquisas futuras.

4.5.2.5. Dominar práticas de desenvolvimento do banco de dados do sistema

O item selecionado, que representa a Competência Técnica 11 (CT11), possui sua distribuição de respostas obtidas ilustrada no Gráfico 7 e a análise feita com base na caracterização dos respondentes disposta no Quadro 9 e Figuras 13 e 14.



1	**1**	2%
2	**5**	9%
3	**23**	43%
4	**16**	30%
5	**8**	15%

Gráfico 7. Dominar práticas de desenvolvimento do banco de dados do sistema (O autor).

Diante dos dados constantes no Quadro 9, é possível visualizar que os respondentes que consideram a competência como “Nada importante” e “Pouco importante” correspondem a 11% do total, sendo seis participantes.

Quadro 9. Análise da CT11 (O autor).

Grau de Importância	**Quantidade de respondentes**		**Membros do time ágil**	
	Abs	**Per**	**Abs**	**Per**
1 e 2	6	11%	6	100%
4 e 5	24	45%	16	67%

Com base na Figura 13, é possível verificar que o perfil preponderante dos respondentes que atribuíram a CT11 como “Nada importante” e “Pouco importante”, que foi destacado de vermelho, é de um profissional que utilizou desenvolvimento ágil de software na área acadêmica, pois quatro deles afirmaram ter utilizado apenas na área acadêmica.

Já os respondentes que consideram a competência como “Muito importante” e “Extremamente Importante” correspondem a 45% do total, sendo 24 participantes.

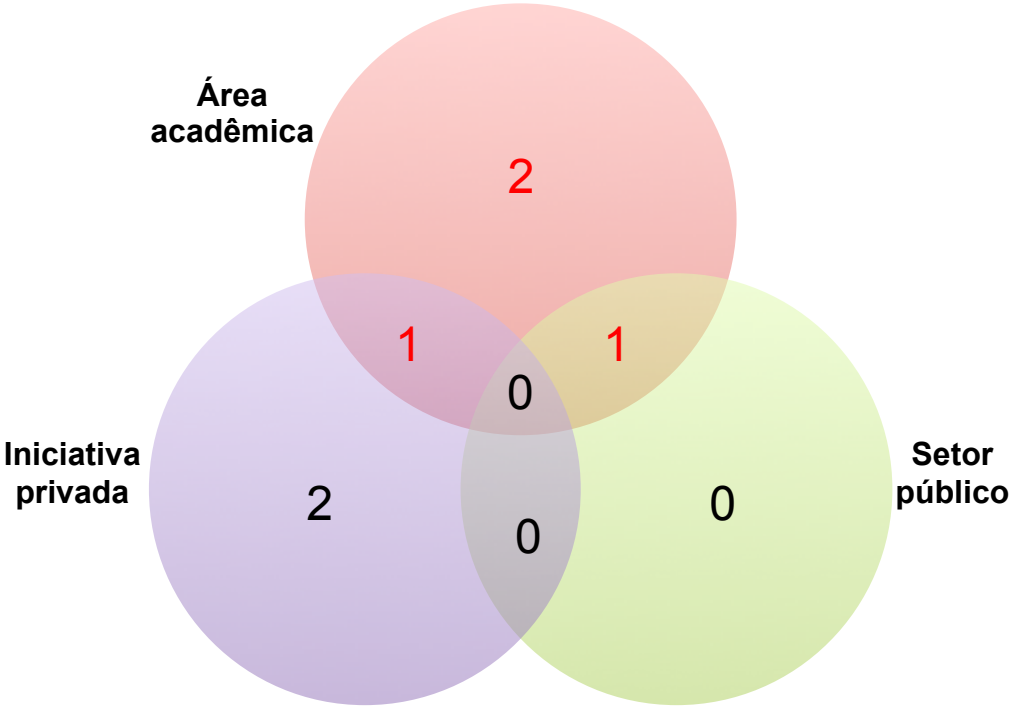


Figura 13. Quantidade de respondentes por contexto para os graus de importância 1 e 2 da CT11 (O autor).

Considerando a Figura 14, é possível verificar que o perfil preponderante dos respondentes que atribuíram a CT11 como “Muito importante” e “Extremamente importante”, que foi destacado de vermelho, é de um profissional que utilizou desenvolvimento ágil de software na área acadêmica, pois oito deles afirmaram ter utilizado apenas na área acadêmica.

É válido ressaltar também que, daqueles que consideraram a competência como “Nada importante” e “Pouco importante”, todos os dois já foram membros de times ágeis; já os que consideraram a competência como “Muito importante” e

“Extremamente Importante”, 21 deles já participaram de times ágeis e fazem parte da maioria que classifica como relevante a CT11.

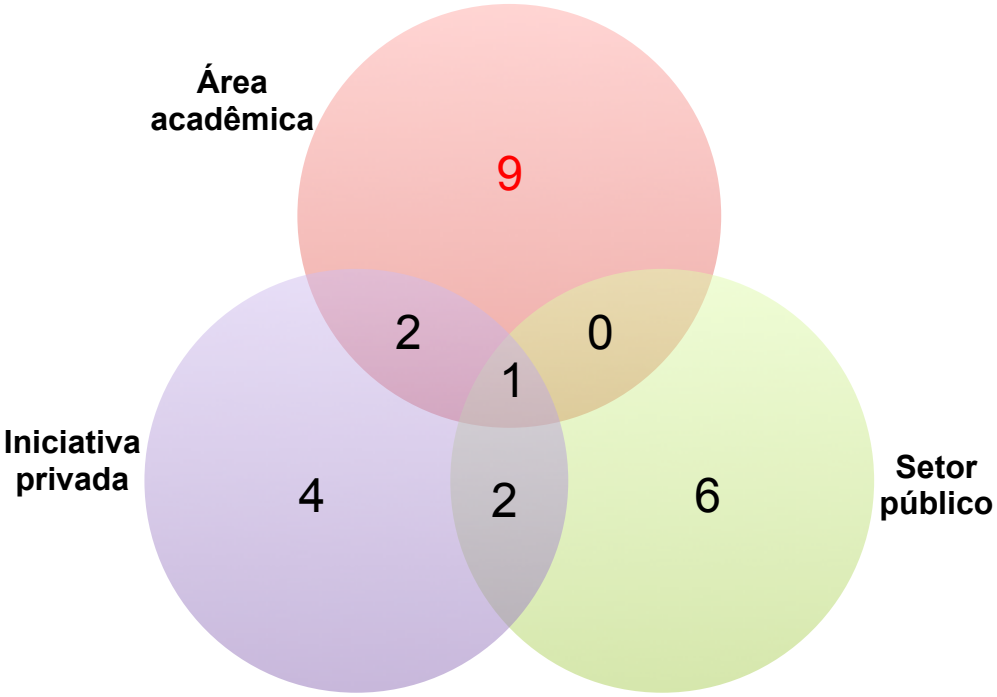


Figura 14. Quantidade de respondentes por contexto para graus de importância 4 e 5 da CT11 (O autor).

A partir dos gráficos, quadros e interpretações feitos, nota-se que a competência “Dominar práticas de desenvolvimento do banco de dados do sistema” (CT11), apesar de apresentar um pico de frequência de respostas na escala “Mediamente importante”, possui a maior parte das respostas nas escalas à direita que expressam a importância da competência, sendo que a maioria dessas respostas vem de membros de times ágeis, o que torna essa competência importante para o contexto do desenvolvimento ágil de software e consistente diante da pesquisa realizada.

4.6. RANKING DE COMPETÊNCIAS

Com o intuito de gerar uma visão panorâmica das competências mais importantes de acordo com a opinião dos participantes dessa pesquisa, foi construído um ranking com todas as competências propostas.

A ordenação das competências se deu da seguinte maneira: como cada item do questionário se relacionava com cada competência que ele contemplava e indagava ao respondente, somou-se a quantidade absoluta da frequência de resposta dos itens em relação aos valores 3, 4 e 5 da escala definida, já que são escalas que indicam ser a competência “Mediamente importante”, “Muito importante” e “Extremamente importante”, respectivamente.

Em casos que ocorreram empate entre competências que tiveram o mesmo valor com a soma absoluta das frequências, o desempate foi feito pela quantidade de respostas obtida para a escala que expressasse maior importância, como por exemplo, o maior número de respostas atribuído à escala “Extremamente importante”. Mesmo assim, houve empate entre competências com mesmo número de resposta, mas foram desempatadas considerando o maior número de respostas obtido para a próxima escala que expressasse maior importância.

Seguindo essa ordenação e os desempates feitos, o ranking foi concebido sendo composto por todas as competências propostas por este trabalho, tanto técnicas quando comportamentais. O ranking está disposto no Quadro 10.

Quadro 10. Ranking das competências (O autor).

Colocação	**ID**	**Competência**	**Resultado**	
			Absoluto	**Percentual**
1	CC1	Capacidade de compartilhar e transferir o conhecimento obtido	52	98%
2	CC12	Capacidade de evolução e aprendizado com o progresso do trabalho	52	98%
3	CC3	Capacidade de trabalhar em equipe e/ou junto com outro membro da equipe	52	98%

Colocação	ID	Competência	Resultado	
			Absoluto	Percentual
4	CC7	Proatividade e iniciativa para a busca de conhecimento e designação de tarefas	52	98%
5	CC5	Flexibilidade e equilíbrio para gerenciar os conflitos do time	52	98%
6	CT8	Dominar a prática de refatoração de código	52	98%
7	CC6	Capacidade de cumprir com seus compromissos	51	96%
8	CC2	Facilidade de comunicação com todos	51	96%
9	CC11	Responsabilidade do time por tudo que foi produzido por ele	51	96%
10	CC13	Capacidade de auto-gerenciar suas atividades e trabalho	51	96%
11	CC10	Visão focada nos objetivos do time e compartilhada com os membros dele	51	96%
12	CT16	Capacidade de estimar esforço de acordo com suas limitações	51	96%
13	CT5	Dominar linguagens de programação	51	96%

Colocação	ID	Competência	Resultado	
			Absoluto	Percentual
14	CC9	Capacidade de tomar decisões diante de situações de incerteza	51	96%
15	CT3	Conhecer os princípios do desenvolvimento ágil de software	50	94%
16	CC4	Disposição para ajudar o time ou membros do time sempre que for necessário	50	94%
17	CC8	Capacidade de estabelecer relacionamentos de confiança com os membros do time	50	94%
18	CT2	Conhecer os valores do desenvolvimento ágil de software	50	94%
19	CT9	Dominar a prática de user stories	50	94%
20	CT4	Conhecer metodologias ágeis de desenvolvimento de software	50	94%
21	CT6	Dominar o uso de padrões de codificação	50	94%
22	CT1	Conhecer a instituição em que trabalha para desempenhar seu papel	50	94%
23	CT10	Dominar práticas de desenvolvimento da arquitetura do sistema	50	94%

Colocação	ID	Competência	Resultado	
			Absoluto	Percentual
24	CT15	Capacidade de negociar o escopo do trabalho	49	92%
25	CT14	Dominar a prática de reuso de software	48	90%
26	CT12	Dominar práticas e padrões de gerência de configuração de software	47	88%
27	CT7	Dominar a prática de TDD e seus padrões	47	88%
28	CT11	Dominar práticas de desenvolvimento do banco de dados do sistema	47	88%
29	CT13	Capacidade de construir e configurar o ambiente de desenvolvimento	46	86%

5. CONSIDERAÇÕES FINAIS

5.1. CONCLUSÕES

Neste Trabalho de Conclusão de Curso objetivou-se definir um conjunto preliminar de competências para um time ágil, que foi feito com base no referencial teórico, na revisão sistemática e no mapeamento das atividades e relacionamentos do papel do time ágil.

Estabelecido e ajustado esse conjunto de competências composto por 29 competências, sendo 13 comportamentais e 16 técnicas, foram feita avaliações quanto ao grau de importância desse conjunto utilizando um instrumento de pesquisa. O resultado observado dessas avaliações foi que 26 das competências apresentaram grau de importância elevado e consistência confirmada e as outras 3 restantes foram analisadas detalhadamente com base no perfil dos respondentes.

Mesmo sem obedecer ao critério de análise, duas das 26 competências bem estabelecidas também foram analisadas para gerar informações adicionais sobre a percepção do desenvolvimento ágil atualmente, onde foram questionados o uso de uma modelagem arquitetural do sistema, ao invés dá metáfora, e a criação de user stories pelo time ágil, ao invés do *Product Owner*.

A análise feita nas competências selecionadas pelo critério descrito neste trabalho constatou que a quantidade de respondentes que optaram pelas escalas que expressam baixo grau de importância foi bem inferior do que a quantidade que optou pelas escalas que refletem alto grau de importância para as competências analisadas, tendo também maior participação de respondentes que foram membros de times ágeis.

Quanto ao perfil dos participantes dessa pesquisa, cabe ressaltar que 77% das respostas são do Distrito Federal, 41,5% do total de respondentes atuam no setor público e a maioria dos participantes (81%) já foi membro de um time ágil.

Diante de todos os dados obtidos e analisados, os resultados são condizentes com a bibliografia explorada, remetendo à consistência dos procedimentos utilizados e dos dados constantes neste documento.

Este trabalho é parte de um projeto de pesquisa sobre competências, realizado na Universidade de Brasília – Campus Gama (UnB/FGA). Este projeto é

contemplado por mais dois trabalhos que se completam e complementam, onde o segundo disposto abaixo ainda se encontra em andamento:

- “Proposta de um conjunto de competências para um *Product Owner*” (BRITO, V. M.);
- “*Proposta de um conjunto de competências para um ScrumMaster*” (QUERUBIM, T.S.).

Por fim, foram observados os pontos fortes e fracos durante e após a pesquisa realizada. Os pontos fortes encontrados foram:

- 81% dos respondentes foram membros de times ágeis, sendo 98% deles satisfeitos com essa experiência;
- As competências avaliadas tiveram um grau de importância tendendo para as escalas mais à direta ou aquelas que expressam mais importância, remetendo à consistência das competências elaboradas e sua concordância com o contexto do trabalho;
- Nenhuma das competências foi considerada como “Nada importante” ou “Pouco importante”;
- O instrumento de pesquisa aplicado apresentou um excelente nível de confiabilidade com base no coeficiente alfa de Cronbach;
- A validação feita com os entrevistados de vários contextos de desenvolvimento de software gerou vários ajustes importantes no questionário e nas competências elaboradas, colaborando nos bons resultados dessa pesquisa;
- Não foram sugeridas e/ou propostas novas competências pelos respondentes na parte de sugestões e críticas do questionário, o que colabora com a completude e abrangência do conjunto de competências elaborado;
- As competências analisadas, apesar de apresentarem um quantitativo maior de respondentes na escala “Mediamente importante” e salvo aquelas bem estabelecidas que foram selecionadas para análise, tiveram um percentual

maior de respondentes para as escalas de mais importância, o que indica a consistência delas.

Já os pontos fracos encontrados, que refletem as fragilidades da pesquisa, foram:

- Curto prazo para disponibilidade do instrumento de pesquisa, impactando no número de respostas;
- Dados coletados pelo instrumento de pesquisa referentes à experiência do respondente não foram informados a unidade de tempo;
- Pouca experiência dos respondentes com desenvolvimento ágil de software, como membro do time ágil e com outros papéis desempenhados.

5.2. TRABALHOS FUTUROS

O prazo de disponibilidade e divulgação do instrumento de pesquisa foi limitado por conta do trabalho estar vinculado a um calendário acadêmico, por isso houve dificuldade para obter mais respostas de pessoas, tanto do Distrito Federal, quanto de outros estados. Com isso, é sugerido que o mesmo instrumento de pesquisa seja aplicada com um prazo maior de disponibilidade e divulgação mais ampla a nível nacional.

Outros trabalhos futuros poderiam estar relacionados ao conjunto de competências proposto, podendo se dar como estudos específicos de uma ou mais competências de maneira isolada, estudo da relação entre as competências e estudos de caso com aplicação delas no contexto real de desenvolvimento software para que seja analisada sua importância, consistência e impacto na prática.

Assim, como sugerido por um dos entrevistados nesse trabalho, é possível também criar uma ferramenta ou um modelo de gestão de competências para times ágeis a partir do que feito neste trabalho, auxiliando na composição de melhores profissionais e na identificação daqueles mais adequados.

Bibliografia

ALLIANCE, S. **"Why Scrum?**. Scrum Alliance, 2013. Disponível em: <www.scrumalliance.org/why-scrum>. Acessado em: 24/09/2013.

AMBLER, S. **Agile modeling: effective practices for extreme programming and the unified process**. John Wiley & Sons, 2002.

BECK, K., BEEDLE, M. COCKBURN, A. et al. **Manifesto para o desenvolvimento ágil de software.** 2001. Disponível em: <www.manifestoagil.com.br>. Acessado em: 20/10/2013.

BECK, K. **Programação Extrema (XP) explicada: Acolha as mudanças**. Bookman, 2004.

BERCZUK, S. **Back to basics: The role of agile principles in success with an distributed scrum team**. In: Agile Conference (AGILE), 2007. IEEE, 2007. p. 382-388.

BLOM, M. **Is Scrum and XP suitable for CSE Development?**. Procedia Computer Science, v. 1, n. 1, p. 1511-1517, 2010.

BRITO, V. M. **Proposta de um conjunto de competências para um *Product Owner*.** Monografia (Graduação) – Universidade de Brasília Faculdade do Gama, Brasília, 2014.

BROOKS JR, F. P. **The Mythical Man-Month, Anniversary Edition: Essays on Software Engineering**. Pearson Education, 1995.

CARLSON, R., TURNER, R. **Review of Agile Case Studies for Applicability to Aircraft Systems Integration**. Procedia Computer Science, v. 16, p. 469-474, 2013.

COCKBURN, A, HIGHSMITH, J. **Agile software development, the people factor**. Computer, v. 34, n. 11, p. 131-133, 2001.

COELHO NETO, V. **Competências organizacionais para o desenvolvimento estratégico do negócio de exploração de petróleo em campos maduros: o caso Petrorecôncavo S.A**. Dissertação (Mestrado Profissional) — Universidade Federal da Bahia. Escola de Administração, Salvador, 2005.

COHN, M. **Succeeding with Agile: Software Development with Scrum**. Addison-Wesley, 2009.

COHN, M. **User Stories Applied: For Agile Software Development**. Addison-Wesley, 2004.

CORAM, M., BOHNER, S. **The impact of agile methods on software project management**. In: Engineering of Computer-Based Systems, 2005. ECBS'05. 12th IEEE International Conference and Workshops on the. IEEE, 2005. p. 363-370.

CORTINA, J. M. **What is coefficient alpha? An examination of theory and applications**. Journal of applied psychology, v. 78, n. 1, p. 98-104, 1993.

CRESPO, A. A.. **Estatística fácil**. Saraiva, 2002.

CRONBACH, L. J. **Coefficient alpha and the internal structure of tests**. Psychometrika, v. 16, n. 3, p. 297-334, 1951.

DORAIRAJ, S., NOBLE, J., MALIK, P. **Understanding lack of trust in distributed agile teams: A grounded theory study**. 2012.

DOWNS, J., HOSKING, J., PLIMMER, B. **Status communication in agile software teams: A case study**. In: Software Engineering Advances (ICSEA), 2010 5th International Conference on. IEEE, 2010. p. 82-87.

DURAND, T. **L'alchimie de la compétence**. Revue française de gestion, n. 1, p. 261-292, 2006.

DYBÅ, T. **Improvisation in Small Software Organizations: Implications for Software Process Improvement**. IEEE Software, v. 17, n. 5, p. 82-87, 2000.

FERRARINI, J. E. A.. **Identificação e valoração de competências para o desenvolvedor de sistemas de informação, na visão dos gestores de fábrica de software de Salvador**. Salvador: Universidade Federal da Bahia, Escola de Administração, 2006.

FINK, A., KOSECOFF, J. **How to Conduct Surveys: A Step by Step Guide**. Sag e Publication Inc, California, 1985.

FITZGERALD, B., HARTNETT, G., CONBOY, K. **Customising agile methods to software practices at Intel Shannon**. European Journal of Information Systems, v. 15, n. 2, p. 200-213, 2006.

FOJTIK, R. **Extreme Programming in development of specific software**. Procedia Computer Science, v. 3, p. 1464-1468, 2011.

FLEURY, A., FLEURY, M. T. L. **Estratégias Empresariais e Formação de Competências: Um quebra-cabeça caleidoscópico da indústria brasileira**. 3a. ed. São Paulo: Atlas, 2004.

FLEURY, M. T. L., FLEURY. A. **Construindo o conceito de competência**. Revista de administração contemporânea, v. 5, n. SPE, p. 183-196, 2001.

FOWLER, M. **The new methodology**. Wuhan University Journal of Natural Sciences, v. 6, n. 1-2, p. 12-24, 2001.

GEORGE, D., MALLERY, P. **SPSS for Windows step by step: A simple guide and reference**. 11.0 update, 4a. ed. Boston: Allyn & Bacon, 2003.

GÜNTHER, H. **Como elaborar um questionário. Instrumentos psicológicos: manual prático de elaboração**, p. 231-258, 1999.

GRAMIGNA, M. R. **Modelo de Competências e Gestão dos Talentos**. São Paulo: Makron Books, 2002.

HAMEL, G., PRAHALAD, C.K. **Competindo pelo futuro: estratégias inovadoras para obter o controle do seu setor e criar os mercados de amanhã**. Rio de Janeiro: Campus, 1995.

HIGHSMITH, J., COCKBURN, A. **Agile software development: The business of innovation. Computer**, v. 34, n. 9, p. 120-127, 2001.

KLIMNIK, Z.M., CASTILHO, I.V., & SANT'ANNA, A.S. **Carreiras em transformação e seus paradoxais reflexos nos indivíduos: Metáforas de carreira e de competências**. In: Comportamento Organizacional e Gestão, 2006, vol. 12, nº 2, p. 257-280.

KOSCIANSKI, A., SOARES, M. S. **Qualidade de software: aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software**. 2. ed. São Paulo: Novatec, 2007.

LE BOTERF, G. **Desenvolvendo a competência dos profissionais**. Porto Alegre: Artmed, 2003.

LEFFINGWELL, Dean. **Agile software requirements: lean requirements practices for teams, programs, and the enterprise**. Addison-Wesley Professional, 2011.

LEME, R. **Avaliação de desempenho com foco em competência: a base para a remuneração por competências.** Rio de Janeiro: Qualitymark, 2006.

LICORISH, S., PHILPOTT, A., MACDONELL, S. G. **Supporting agile team composition: A prototype tool for identifying personality (in) compatibilities**. In: Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering. IEEE Computer Society, 2009. p. 66-73.

MCHUGH, O., CONBOY, K., LANG, M. **Agile practices: The impact on trust in software project teams**. Software, IEEE, v. 29, n. 3, p. 71-76, 2012.

MOE, N. B., DINGSØYR, T., DYBÅ, T. **A teamwork model for understanding an agile team: A case study of a Scrum project**. Information and Software Technology, v. 52, n. 5, p. 480-491, 2010.

NERUR, S., BALIJEPALLY, V. **Theoretical reflections on agile development methodologies**. Communications of the ACM, v. 50, n. 3, p. 79-83, 2007.

PARRY, S. B. **The quest for competencies**. Training Magazine, v. 33, n. 7, Julho 1996.

PICARELLI Filho, V., WOOD Jr., T. **Remuneração por habilidades e por competências: preparando a organização para a era das empresas de conhecimento intensivo**. 2. ed. rev Sao Paulo: Atlas, 1999.

PRESSMAN, R. S. **Engenharia de software**. McGraw Hill Brasil, 1995.

RESENDE, E. **O Livro das Competências: Desenvolvimento das competências**. 2a. ed. São Paulo: Editora Qualitymark, 2003.

RUHNOW, A. **Consciously evolving an agile team**. Agile Conference, IEEE, 2007. p. 130-135.

SCHUMAN, H., KALTON, G. **Survey methods**. Handbook of social psychology, v. 1, p. 635-697, 1985.

SCHWABER, K., SUTHERLAND, J. **The Scrum guide - The definitive guide to Scrum: The rules of the game**. 2013. Disponível em: <www.scrum.org>. Acessado em: 20/09/2013.

SCHWABER, K. **Agile Project Management with Scrum**. Microsoft Press, 2004.

SHORE, J. **A arte do desenvolvimento ágil**. Alta Books, 2008.

STOTT, W., NEWKIRK, J. **Visual studio team system: better software development for agile teams**. Addison-Wesley Professional, 2007.

STREINER, D. L. **Being inconsistent about consistency: when coefficient alpha does and doesn´t matter**. Journal of Personality Assessment. v. 80, p. 217-222. 2003.

TAKEUCHI, H., NONAKA, I. **The new new product development game**. Harvard business review, v. 64, n. 1, p. 137-146, 1986.

TELES, V. **Extreme Programming: Aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade**. São Paulo, Novatec Editora, 2006.

TENGSHE, A., NOBLE, S. **Establishing the agile PMO: Managing variability across projects and portfolios**. In: Agile Conference (AGILE), 2007. IEEE, 2007. p. 188-193.

TSIRAKIDIS, P., KOBLER, F., KRCMAR, H. **Identification of success and failure factors of two agile software development teams in an open source organization**. In: Global Software Engineering, 2009. ICGSE 2009. 4th IEEE International Conference on. IEEE, 2009. p. 295-296.

WALPOLE, R. E. **Probabilidade & Estatística para engenharia e ciências**. Pearson Prentice Hall, 2009.

WOOD, S., MICHAELIDES, G., THOMSON, C. **Successful extreme programming: Fidelity to the methodology or good teamworking?**. Information and Software Technology, v. 55, n. 4, p. 660-672, 2013.

YAREMKO, R. K., HARARI, H., HARRISON, R. C., & Lynn, E. **Handbook of research and quantitative methods in psychology**. Hillsdale, NJ: Lawrence Erlbaum, 1986.

ZARIFIAN, P. **Objetivo competência: por uma nova lógica**. São Paulo: Atlas, 2001.

APÊNDICE I – PROTOCOLO DA REVISÃO SISTEMÁTICA

1. **Objetivo –** Identificar as características de times ágeis.
2. **Questão principal –** Quais as características ou propriedades de times ágeis?
    a. **População** – Projetos de software que utilizam metodologias ágeis.
    b. **Intervenção** – Times ágeis.
    c. **Resultados** – Uma lista de características de times ágeis.
    d. **Aplicação** - Servir de base ou apoiar pesquisas que irão tratar sobre times ágeis.
3. **Fontes de pesquisa:** ACM Digital Library, IEEEXplore e ScienceDirect.
4. **Critérios de seleção -** Os critérios de inclusão e exclusão foram aplicados no título, abstract e índice dos trabalhos encontrados.
    a. **Inclusão:** Documento está disponível na internet, fala sobre times ágeis e apresenta estudo de caso.
    b. **Exclusão:** Documento não está disponível na internet, não fala sobre times ágeis, não possui estrutura adequada de um trabalho de pesquisa e documento trata de integração de times ágeis com conceitos fora do contexto de desenvolvimento ágil.
5. **Palavras-chave -** agile, engineering, high, managing, performance, performing, project, self, software, team, team-work, teamwork, work, practice, technique.
6. **String de busca** –
    a. **ACM:** (Title:agile) and (Title:team or Title:teamwork or Title:self or Title:organizing or Title:managing or Title:high or Title:performing or Title:performance) and (Title:software or Title:project or Title:engineering)
    b. **IEEE:** (("Document Title":"agile") AND ("Document Title":"team" OR "work" or "teamwork" OR "team-work" OR "self" OR "organizing" OR

"managing" OR "high" OR "performing" OR "performance") AND ("Document Title":"software" OR "project" OR "engineering")).

c. **ScienceDirect:** agile and team and (xp or scrum or lean or rapid or crystal) and (practice or technique)[All Sources(Business, Management and Accounting,Computer Science,Engineering)]

APÊNDICE II – NÍVEL DE FUNDAMENTAÇÃO DAS CARACTERÍSTICAS DO TIME ÁGIL

Quadro 11. Nível de fundamentação das características por artigo.

Característica	Referências	Disposição
Confiança	(MCHUGH, et. al., 2012)	Citada, explicada e evidente em estudo de caso.
	(MOE, et. al., 2010)	Citada, explicada e evidente em estudo de caso.
	(TSIRAKIDIS, et. al., 2009)	Apenas citada.
	(STOTT e NEWKIRK, 2007)	Apenas citada.
	(DORAIRAJ, et. al., 2012)	Citada, explicada e evidente em estudo de caso.
	(WOOD, et. al., 2013)	Citada, explicada e evidente em estudo de caso.
Comprometimento	(MCHUGH, et. al., 2012)	Apenas citada.
	(TENGSHE e NOBLE, 2007)	Citada e explicada.
	(DOWNS, et. al., 2010)	Apenas citada.
	(BERCZUK, 2007)	Apenas citada.
	(DORAIRAJ, et. al., 2012)	Citada, explicada e evidente em estudo de caso.
	(CARLSON e TURNER, 2013)	Citada.
Comunicação constante	(MCHUGH, et. al., 2012)	Citada e explicada.
	(MOE, et. al., 2010)	Citada, explicada e evidente em estudo de caso
	(TENGSHE e NOBLE, 2007)	Apenas citada.
	(TSIRAKIDIS, et. al., 2009)	Apenas citada.
	(DOWNS, et. al., 2010)	Citada, explicada e evidente em estudo de caso
	(STOTT e NEWKIRK, 2007)	Citada, explicada e evidente em estudo de caso
	(BERCZUK, 2007)	Citada, explicada e evidente em estudo de caso
	(RUHNOW, 2007)	Citada e explicada.
	(LICORISH, et. al., 2009)	Apenas citada.
	(CORAM e BOHNER, 2005)	Citada e explicada.
	(DORAIRAJ, et. al., 2012)	Citada, explicada e evidente em estudo de

Característica	Referências	Disposição
		caso.
	(FOJTIK, 2011)	Citada e explicada.
	(WOOD, et. al., 2013)	Citada e explicada.
	(BLOM, 2010)	Citada e explicada.
Compartilhamento de conhecimento	(MCHUGH, et. al., 2012)	Citada e explicada.
	(MOE, et. al., 2010)	Apenas citada.
	(TSIRAKIDIS, et. al., 2009)	Apenas citada.
	(DOWNS, et. al., 2010)	Citada e explicada.
	(RUHNOW, 2007)	Citada e explicada.
	(LICORISH, et. al., 2009)	Apenas citada.
	(CORAM e BOHNER, 2005)	Apenas citada.
	(CARLSON e TURNER, 2013)	Citada, explicada e evidente em estudo de caso.
	(WOOD, et. al., 2013)	Citada, explicada e evidente em estudo de caso.
	(BLOM, 2010)	Citada, explicada e evidente em estudo de caso.
Responsabilidade coletiva	(MCHUGH, et. al., 2012)	Citada e explicada.
	(MOE, et. al., 2010)	Citada e explicada.
	(DOWNS, et. al., 2010)	Citada e explicada.
	(STOTT e NEWKIRK, 2007)	Citada e explicada.
	(RUHNOW, 2007)	Citada e explicada.
	(WOOD, et. al., 2013)	Citada e explicada.
	(BLOM, 2010)	Citada e explicada.
Uso de Programação em pares	(MCHUGH, et. al., 2012)	Apenas citada.
	(MOE, et. al., 2010)	Apenas citada.
	(TSIRAKIDIS, et. al., 2009)	Apenas citada.
	(STOTT e NEWKIRK, 2007)	Citada, explicada e evidente em estudo de caso.
	(RUHNOW, 2007)	Citada, explicada e evidente em estudo de caso.
	(CORAM e BOHNER, 2005)	Citada e explicada.
	(FOJTIK, 2011)	Citada e explicada.
	(BLOM, 2010)	Citada e explicada.
Colaboração	(MOE, et. al., 2010)	Citada, explicada e evidente em estudo de caso.
	(TENGSHE e NOBLE, 2007)	Apenas citada.
	(STOTT e NEWKIRK, 2007)	Apenas citada.
	(RUHNOW, 2007)	Citada e explicada.
	(LICORISH, et. al., 2009)	Apenas citada.
	(CORAM e BOHNER, 2005)	Citada e explicada.

Característica	Referências	Disposição
	(DORAIRAJ, et. al., 2012)	Citada, explicada e evidente em estudo de caso.
	(WOOD, et. al., 2013)	Citada e explicada.
Modelo mental compartilhado	(MOE, et. al., 2010)	Citada, explicada e evidente em estudo de caso.
	(TSIRAKIDIS, et. al., 2009)	Apenas citada.
Autonomia	(MOE, et. al., 2010)	Apenas citada.
	(TENGSHE e NOBLE, 2007)	Apenas citada.
	(STOTT e NEWKIRK, 2007)	Apenas citada.
	(CORAM e BOHNER, 2005)	Apenas citada.
	(CARLSON e TURNER, 2013)	Citada, explicada e evidente em estudo de caso.
	(WOOD, et. al., 2013)	Citada, explicada e evidente em estudo de caso.
Melhoria contínua	(TENGSHE e NOBLE, 2007)	Apenas citada.
	(BERCZUK, 2007)	Apenas citada.
	(RUHNOW, 2007)	Citada e explicada.
	(CARLSON e TURNER, 2013)	Citada, explicada e evidente em estudo de caso.
	(BLOM, 2010)	Citada, explicada e evidente em estudo de caso.
Uso de Desenvolvimento orientado a testes	(TSIRAKIDIS, et. al., 2009)	Citada e explicada.
	(DOWNS, et. al., 2010)	Citada e explicada.
	(STOTT e NEWKIRK, 2007)	Citada, explicada e evidente em estudo de caso.
	(BERCZUK, 2007)	Citada e explicada.
	(RUHNOW, 2007)	Citada, explicada e evidente em estudo de caso.
	(CORAM e BOHNER, 2005)	Citada e explicada.
	(FOJTIK, 2011)	Citada e explicada.
	(WOOD, 2013)	Apenas citada.
	(BLOM, 2010)	Citada, explicada e evidente em estudo de caso.
Uso de Refatoração	(TSIRAKIDIS, et. al., 2009)	Apenas citada.
	(STOTT e NEWKIRK, 2007)	Citada, explicada e evidente em estudo de caso.
	(RUHNOW, 2007)	Apenas citada.
	(CORAM e BOHNER, 2005)	Citada e explicada.

Característica	Referências	Disposição
	(WOOD, et. al., 2013)	Apenas citada.
Proatividade	(DOWNS, et. al., 2010)	Apenas citada.
Uso de Integração contínua	(DOWNS, et. al., 2010)	Citada e explicada.
	(STOTT e NEWKIRK, 2007)	Citada, explicada e evidente em estudo de caso.
	(BERCZUK, 2007)	Citada e explicada.
	(RUHNOW, 2007)	Citada, explicada e evidente em estudo de caso.
	(CORAM e BOHNER, 2005)	Citada e explicada.
	(CARLSON e TURNER, 2013)	Citada, explicada e evidente em estudo de caso.
	(WOOD, et. al., 2013)	Citada e explicada.
	(BLOM, 2010)	Citada e explicada.
Uso de Controle de versão	(STOTT e NEWKIRK, 2007)	Citada, explicada e evidente em estudo de caso.
Solucionar conflitos	(RUHNOW, 2007)	Citada e explicada
	(DORAIRAJ, et. al., 2012)	Citada, explicada e evidente em estudo de caso.
Uso de Users Stories	(TENGSHE e NOBLE, 2007)	Citada, explicada e evidente em estudo de caso.
	(STOTT e NEWKIRK, 2007)	Citada, explicada e evidente em estudo de caso.
	(RUHNOW, 2007)	Citada, explicada e evidente em estudo de caso.
	(CORAM e BOHNER, 2005)	Apenas citada.
	(CARLSON e TURNER, 2013)	Citada, explicada e evidente em estudo de caso.
Uso de Padrões de código	(TSIRAKIDIS, et. al., 2009)	Apenas citada.
	(STOTT e NEWKIRK, 2007)	Citada, explicada e evidente em estudo de caso.
	(WOOD, et. al., 2013)	Citada e explicada.
	(BLOM, 2010)	Citada, explicada e evidente em estudo de caso.

APÊNDICE III – CONJUNTO PRELIMINAR DE COMPETÊNCIAS DO TIME ÁGIL

Quadro 12. Competências comportamentais de um time ágil.

ID	Competências comportamentais
CC1	Capacidade de compartilhar e transferir o conhecimento obtido
CC2	Facilidade de comunicação com todos os níveis da organização
CC3	Capacidade de trabalhar em equipe e/ou junto com outro membro da equipe
CC4	Disposição para ajudar o time ou membros do time sempre que for necessário
CC5	Flexibilidade e equilíbrio para gerenciar os conflitos do time
CC6	Zelo e empenho pelo cumprimento das atividades e obrigações do time
CC7	Proatividade e iniciativa para a busca de conhecimento e designação de tarefas
CC8	Capacidade de estabelecer relacionamentos de confiança com os membros do time
CC9	Capacidade de tomar decisões diante de situações de incerteza
CC10	Visão focada nos objetivos do time e compartilhada com os membros dele
CC11	Responsabilidade pelas atividades e tarefas do time como um todo e por tudo que foi produzido por ele
CC12	Capacidade de evolução e aprendizado com o progresso do trabalho
CC13	Capacidade de auto-gerenciar suas atividades e trabalho

Quadro 13. Competências técnicas de um time ágil.

ID	Competências técnicas
CT1	Conhecer a instituição em que trabalha para desempenhar seu papel
CT2	Conhecer metodologias ágeis de desenvolvimento de software

ID	Competências técnicas
CT3	Dominar linguagens de programação
CT4	Dominar técnicas de programação
CT5	Dominar a prática de TDD e seus padrões
CT6	Dominar a prática de Integração contínua de software
CT7	Dominar a prática de refatoração de código
CT8	Dominar prática de User story
CT9	Dominar práticas de desenvolvimento da arquitetura do sistema
CT10	Dominar práticas de desenvolvimento do banco de dados do sistema
CT11	Dominar práticas e padrões de gerência de configuração de software
CT12	Capacidade de construir e configurar o ambiente de desenvolvimento
CT13	Dominar a prática de reuso de software

APÊNDICE IV – PRIMEIRA VERSÃO DO QUESTIONÁRIO

Editar este formulário

Competências de Times ágeis

Caro(a) colega,

Esta pesquisa faz parte de um Trabalho de Conclusão de Curso, de graduação em Engenharia de Software, pela Universidade de Brasília (UnB). Ela visa identificar a percepção de profissionais, estudantes e acadêmicos de Engenharia de Software, mais especificamente, em desenvolvimento de software utilizando metodologias ágeis, sobre o grau de importância de cada uma das competências (técnicas e comportamentais) que um time ágil deve possuir.

As competências são avaliadas pelo seu grau de importância que varia de 1 a 5, sendo:

1: Nada importante
2: Pouco importante
3: Mediamente importante
4: Muito importante
5: Extremamente importante

As competências técnicas presentes neste formulário estão relacionadas aos conhecimentos e habilidades de um time ágil e seus membros, enquanto as comportamentais estão relacionadas a suas atitudes.

Cada assertiva será identificada por siglas que definem qual é o tipo da competência em questão.

CC - Competência comportamental
CT - Competência técnica

A partir de sua experiência e conhecimento em metodologias ágeis de desenvolvimento de software, julgue e atribua o grau de importância para cada competência.

Idealmente e conforme pesquisado e descrito no Trabalho de Conclusão de Curso, um time ágil deve possuir as seguintes competências:

*Obrigatório

1) CC - Capacidade de compartilhar e transferir o conhecimento obtido. *
Esta competência comportamental está relacionada ao compartilhamento de conhecimento que ocorre em times ágeis, o que remete às reuniões ou eventos que necessitam informar o andamento do trabalho ou instruir o membro do time em alguma dúvida ou dificuldade.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

2) CC - Facilidade de comunicação com todos os níveis da organização. *
Esta competência comportamental está relacionada à comunicação constante em times ágeis necessária para que eles consigam lidar e se articular com os stakeholders e entre seus membros, relatando o que está sendo feito e favorecendo a transparência do projeto.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

3) CC - Capacidade de trabalhar em time e/ou junto com outro membro do time. *
Esta competência comportamental está relacionada à prática de programação em pares de times ágeis e à facilidade de seus membros de se adequar em um trabalho em equipe, onde a opinião de todos é relevante.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

4) CC - Disposição para ajudar o time ou membros do time sempre que for necessário. *
Esta competência comportamental está relacionada à colaboração e proatividade evidentes em times ágeis que reflete a importância da participação ativa de todos os membros do time, seja trazendo novas idéias ou auxiliando aqueles que precisam de ajuda.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

5) CC - Flexibilidade e equilíbrio para gerenciar os conflitos do time. *
Esta competência comportamental está relacionada à facilidade de solucionar conflitos evidente em times ágeis, fazendo com que a comunicação seja mais produtiva e as decisões do projeto sejam mais conscientes e consensuais.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

6) CC - Zelo e empenho pelo cumprimento das atividades e obrigações do time. *
Esta competência comportamental está relacionada ao comprometimento do time ágil com suas tarefas, retratando a seriedade dele pela completude das suas atividades.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

7) CC - Proatividade e iniciativa para a busca de conhecimento e designação de tarefas. *
Esta competência comportamental está relacionada à proatividade evidente em times ágeis, refletindo o fato dos membros do time ágil manifestarem interesse e se responsabilizarem por atividades que possuem afinidade e procurarem conhecimento, sempre que necessário, com os stakeholders ou com outros membros do time.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

8) CC - Capacidade de estabelecer relacionamentos de confiança com os membros do time. *
Esta competência comportamental está relacionada à confiança evidente em times ágeis, o que remete à importância da união do time para lidar o trabalho a ser realizado.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

9) CC - Capacidade de tomar decisões diante as situações de incerteza. *
Esta competência comportamental está relacionada à autonomia e proatividade evidentes em times ágeis. O modo como uma user storie ou atividade será feita é uma decisão exclusiva de seu responsável, cabendo a ele relatar ao time o que será feito.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

10) CC - Visão focada nos objetivos do time e compartilhada com os membros dele. *
Esta competência comportamental está relacionada ao modelo mental compartilhado evidente em times ágeis, refletindo a necessidade do time evitar trabalhos extras e desnecessários e propagar isso aos seus membros.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

11) CC - Responsabilidade pelas atividades e tarefas do time como um todo e por tudo que foi produzido por ele. *
Esta competência comportamental está relacionada à responsabilidade coletiva e comprometimento evidentes em times ágeis, pois atividades que não foram concluídas configuram uma imagem negativa ao time inteiro, não apenas aos seus responsáveis.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

12) CC - Capacidade de evolução e aprendizado com o progresso do trabalho. *
Esta competência comportamental está relacionada à tendência de times ágeis melhorarem continuamente. O time ágil procura, através das experiências em iterações passadas e trabalhos realizados, evitar gargalos dentro do processo e atingir excelência técnica para aprimorar a qualidade do seu trabalho.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

13) CC - Capacidade de auto-gerenciar suas atividades e trabalho. *
Esta competência comportamental está relacionada à essência das metodologias ágeis e, principalmente, à característica de times ágeis de se auto-organizar e gerenciar sem a necessidade de um gestor ou gerente.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

1) CT - Conhecer a instituição em que trabalha para desempenhar seu papel. *
Esta competência técnica se refere ao conhecimento que o time ágil tem do contexto no qual ele está inserido para que ele possa melhorar seu planejamento e relacionamentos dentro e fora do time.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

2) CT - Conhecer metodologias ágeis de desenvolvimento de software. *
Esta competência técnica se refere ao conhecimento que o time ágil tem quanto à(s) metodologia(s) ágil(ágeis) que ele utiliza. Ressaltando que a metodologia faz menção ao processo, fases, regras e artefatos que o time ágil segue e utiliza.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

3) CT - Dominar linguagens de programação. *
Esta competência técnica se refere ao domínio em linguagens de programação do time ágil necessário para que ele possa implementar as funcionalidades do sistema.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

4) CT - Dominar técnicas de programação. *
Esta competência técnica se refere ao domínio em técnicas de programação do time ágil como, por exemplo, o uso de padrões de código, que permite ao membro do time uma melhor interpretação do código feito por outro membro e evita que módulos do sistema fiquem restritos a quem os fez.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

5) CT - Dominar a prática de TDD e seus padrões. *
Esta competência técnica está relacionada ao desenvolvimento orientado a teste evidente em times ágeis que concede ao projeto uma cobertura de testes confiável e código enxuto.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

6) CT - Dominar a prática de Integração contínua de software. *
Esta competência técnica está relacionada ao uso de Integração contínua evidente em times ágeis, permitindo ao time ter um ambiente de desenvolvimento atualizado e contendo todas as alterações feitas pelos seus membros.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

7) CT - Dominar a prática de Refatoração de código. *
Esta competência técnica está relacionada à refatoração de código realizada por times ágeis que fornece ao time uma arquitetura do sistema mais coesa e com uma melhor performance.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

8) CT - Dominar a prática de User Stories. *
Esta competência técnica está relacionada ao uso de User Stories por times ágeis como uma maneira mais prática e simples de estimar, planejar e priorizar suas tarefas.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

9) CT - Dominar práticas de desenvolvimento da Arquitetura do sistema. *
Esta competência técnica está relacionada às práticas que envolvem a criação e modelagem da arquitetura a partir das users stories priorizadas pelos stakeholders e modificação dela conforme necessidade do projeto.

	1	2	3	4	5	
Nada importante	◯	◯	◯	◯	◯	Muito importante

10) CT - Dominar práticas de desenvolvimento do Banco de dados do sistema. *
Esta competência técnica está relacionada às práticas que envolvem a criação e modelagem de um banco de dados físico que persistirá as informações e dados do projeto.

	1	2	3	4	5	
Nada importante	◯	◯	◯	◯	◯	Muito importante

11) CT - Dominar práticas e padrões de Gerência de Configuração de software. *
Esta competência técnica está relacionada às práticas de controle de versão, controle de mudança e manuseio de build que são importantes para auxiliar na qualidade do projeto desenvolvido.

	1	2	3	4	5	
Nada importante	◯	◯	◯	◯	◯	Muito importante

12) CT - Capacidade de construir e desenvolver a configuração do ambiente de desenvolvimento. *
Esta competência técnica está relacionada à construção e manutenção da infra-estrutura necessária para comportar o desenvolvimento do projeto.

	1	2	3	4	5	
Nada importante	◯	◯	◯	◯	◯	Muito importante

13) CT - Dominar a prática de reuso de software. *
Esta competência técnica está relacionada à prática de reuso de software que aproveita componentes de outros projetos realizados e os utiliza em projetos atuais. Essa prática colabora com o custo do projeto e facilita o desenvolvimento dele.

	1	2	3	4	5	
Nada importante	◯	◯	◯	◯	◯	Muito importante

Observações

Na sua opinião sobre o desenvolvimento ágil de software, há alguma competência ou tópico importante que não consta neste formulário e deveria estar presente?
Caso tenha, informe no campo abaixo.

Considerações finais

Concluindo este questionário, gostaríamos de fazer algumas perguntas que permitirão melhor caracterizar o grupo de pessoas que participaram dessa pesquisa. Lembramos que todas as suas declarações serão tratadas de maneira confidencial. Os resultados serão apresentados de maneira a não permitir a identificação de participantes individuais.

Em que estado você mora? *

Qual seu grau de formação atual?

- ◯ Superior incompleto
- ◯ Superior completo
- ◯ Pós-graduado
- ◯ Mestrado
- ◯ Doutorado

Há quanto tempo você utiliza/utilizou metodologias ágeis? *

Qual(is) o(s) papel(éis) desempenhado(s) nesse tempo? *

- ◯ ScrumMaster
- ◯ Product Owner
- ◯ Time
- ◯ Stakeholder (Interessado)
- ◯ Patrocionador
- ◯ Estudante
- ◯ Outro:

Qual(is) a(s) aplicação(ões) dessa experiência? *

- ◯ Servidor público
- ◯ Iniciativa privada
- ◯ Acadêmica

Caso a aplicação tenha sido como servidor público, em qual área foi dada essa experiência? *
Caso não tenha sido marcada opção "Servidor público" na pergunta acima, marque a resposta "Não se aplica".

- ◯ Legistalivo
- ◯ Judiciário

- ◯ Executivo
- ◯ Não se aplica

Atualmente, qual sua área de atuação no desenvolvimento de software? *

- ◯ Servidor público
- ◯ Iniciativa privada
- ◯ Acadêmica

Enviar

Nunca envie senhas em Formulários Google.

100% concluído.

APÊNDICE V – SEGUNDA VERSÃO DO QUESTIONÁRIO

Editar este formulário

Competências de Times Ágeis

Caro participante,

Você está sendo convidado(a) a responder um questionário que visa identificar o grau de importância de competências técnicas (CT) e comportamentais (CC) que um TIME ÁGIL de desenvolvimento de software, deve possuir.
O presente questionário é parte de um trabalho de pesquisa do curso de Engenharia de Software da Universidade de Brasília (UnB).

Para esta pesquisa, entende-se por TIME ÁGIL:
"A equipe de profissionais que realizam o trabalho de entregar uma versão usável que potencialmente incrementa o produto “Pronto” ao final de cada Sprint. Somente integrantes do time criam incrementos."

O questionário está estruturado em assertivas para as quais você deverá atribuir uma nota de um a cinco para indicar o grau de importância das competências, de acordo com a escala abaixo:

1: Nada importante
2: Pouco importante
3: Mediamente importante
4: Muito importante
5: Extremamente importante

Lembre-se ainda que:

- O questionário deverá ser respondido individualmente;
- Não existem respostas “certas” ou “erradas”. O importante é mostrar de forma sincera como você percebe cada uma das competências apresentadas;
- Suas respostas serão tabuladas e transformadas em dados, mantido o seu anonimato.

Desde já agradecemos à atenção dispensada e nos colocamos à disposição para os esclarecimentos necessários.

*Obrigatório

Competências comportamentais

Competências que expressam valores, interesses e atitudes na prática profissional.

1) CC - Capacidade de compartilhar e transferir o conhecimento obtido. *
Compartilhar experiências e transferir o que foi aprendido entre o time ágil, seja tanto nas reuniões ou eventos que ele participa quanto diariamente.

	1	2	3	4	5	
Nada importante	◯	◯	◯	◯	◯	Muito importante

2) CC - Facilidade de comunicação com todos. *
Interagir e se articular com todos os comprometidos com o projeto como, por exemplo, o Product Owner, Scrum Master e o próprio time ágil.

	1	2	3	4	5	
Nada importante	◯	◯	◯	◯	◯	Muito importante

3) CC - Capacidade de trabalhar em equipe e/ou junto com outro membro da equipe. *
Facilidade de se adequar ao trabalho em grupo e à prática de programação em pares, onde dois membros do time ágil programam juntos na mesma baia.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

4) CC - Disposição para ajudar o time ou membros do time sempre que for necessário. *
Participação ativa e colaborativa de todos os membros do time ágil tanto para realizar o trabalho quanto para solucionar dúvidas ou dificuldades entre eles.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

5) CC - Flexibilidade e equilíbrio para gerenciar os conflitos do time. *
Saber lidar com conflitos para que sejam melhor solucionados ou contornados.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

6) CC - Capacidade de cumprir com seus compromissos. *
Honrar os compromissos feitos pelo time quanto aos objetivos, às entregas e aos padrões de qualidade do projeto.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

7) CC - Proatividade e iniciativa para a busca de conhecimento e designação de tarefas. *
Manifestar interesse e responsabilidade por atividades que possui afinidade e procurar conhecimento, sempre que necessário, com o Product Owner e com outros membros do time.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

8) CC - Capacidade de estabelecer relacionamentos de confiança com os membros do time. *
Criar e manter relacionamentos transparentes e maduros, pautados na sinceridade e responsabilidade.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

9) CC - Capacidade de tomar decisões diante de situações de incerteza. *
Ter autonomia suficiente para determinar, a nível técnico, a maneira que o projeto será implementado como, por exemplo, quando o responsável por uma user story define como ela será feita e reporta ao time.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

10) CC - Visão focada nos objetivos do time e compartilhada com os membros dele. *
Ter modelo mental compartilhado entre todos os membros do time para evitar trabalhos extras e desnecessários.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

11) CC - Responsabilidade do time por tudo que foi produzido por ele. *
Utilizar a prática de propriedade coletiva onde tudo o que é feito pelos membros do time pertence a todo time, não apenas ao seu responsável.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

12) CC - Capacidade de evolução e aprendizado com o progresso do trabalho. *
Procurar, através das experiências em iterações passadas e trabalhos realizados, melhorar continuamente, evitar gargalos dentro do processo e atingir a excelência técnica.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

13) CC - Capacidade de auto-gerenciar suas atividades e trabalho. *
Organizar e gerenciar o trabalho que está sendo realizado e o que será feito sem a necessidade da figura de um gestor ou gerente.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

Competências técnicas

Todos os conhecimentos e habilidades que o indivíduo precisa ter para o desempenho efetivo de sua atividade profissional.

1) CT - Conhecer a instituição em que trabalha para desempenhar seu papel. *
Conhecer as práticas e normas utilizadas pelo time e as pessoas envolvidas no projeto como, por exemplo, os membros do time, o Product Owner e o Scrum Master.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

2) CT - Conhecer os valores do desenvolvimento ágil de software. *
Ter conhecimento dos valores difundidos pelo Manifesto ágil como, por exemplo, "indivíduos e interações entre eles" e "responder a mudanças".

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

3) CT - Conhecer os princípios do desenvolvimento ágil de software. *
Ter conhecimento dos princípios difundidos pelo Manifesto ágil como, por exemplo, "Contínua atenção à excelência técnica e bom design aumenta a agilidade" e "As melhores arquiteturas, requisitos e designs emergem de equipes auto-organizáveis".

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

4) CT - Conhecer metodologias ágeis de desenvolvimento de software. *
Ter conhecimento do processo, cerimônias, regras e artefatos que o time ágil segue e utiliza.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

5) CT - Dominar linguagens de programação. *
Ter conhecimento e habilidade com linguagens de programação, como, por exemplo, Java e C/C++.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

6) CT - Dominar o uso de padrões de codificação. *
Ser capaz de escrever o código utilizando padrões de escrita e identação, visando facilitar a interpretação e manutenção por outro membro do time.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

7) CT - Dominar a prática de TDD e seus padrões. *
Ser capaz de escrever testes antes do código e cumpri-los com o mínimo de código possível.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

8) CT - Dominar a prática de refatoração de código. *
Ser capaz de modificar um sistema de software de modo que não altere o comportamento externamente observável e ainda melhore sua estrutura interna.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

9) CT - Dominar a prática de user stories. *

Ser capaz de descrever funcionalidades valiosas para os usuário de um sistema, sendo compostas por um breve descrição, conversas ou debates e critérios de aceitação.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

10) CT - Dominar práticas de desenvolvimento da arquitetura do sistema. *

Ser capaz de definir o framework ou estrutura do sistema, seu comportamento, suas propriedades e relacionamentos com outros sistemas.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

11) CT - Dominar práticas de desenvolvimento do banco de dados do sistema. *

Ser capaz de criar e modelar um banco de dados SQL, por exemplo.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

12) CT - Dominar práticas e padrões de gerência de configuração de software. *

Ser capaz de manter a integridade da configuração através de práticas como integração de contínua e controle de versão.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

13) CT - Capacidade de construir e configurar o ambiente de desenvolvimento. *

Ser capaz de construir e manter a infraestrutura necessária para comportar o desenvolvimento do projeto.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

14) CT - Dominar a prática de reuso de software. *

Ser capaz de aproveitar componentes de sistemas finalizados para a construção de um outro novo e diferente.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

15) CT - Capacidade de negociar o escopo do trabalho. *

Ser capaz de deliberar e ponderar sobre o escopo do projeto que será trabalhado para não sobrecarregar o time ágil.

1 2 3 4 5

Nada importante ◯ ◯ ◯ ◯ ◯ Muito importante

16) CT - Capacidade de estimar esforço de acordo com suas limitações. *
Ser capaz de estimar o esforço para realizar as users stories adequadamente considerando as limitações técnicas e físicas do time ágil.

	1	2	3	4	5	
Nada importante	◯	◯	◯	◯	◯	Muito importante

Observações

Na sua opinião sobre o desenvolvimento ágil de software, há alguma competência ou tópico importante que não consta neste formulário e deveria estar presente? Sugestões e críticas também podem ser expressas.
Caso tenha, informe no campo abaixo.

Dados do Respondente

Sexo *
- ◯ Masculino
- ◯ Feminino

Qual sua experiência (tempo) no desenvolvimento ágil de software? *
Tempo em anos e/ou meses. Exemplos: 1 ano e 2 meses, 7 meses, 4 anos.

Já foi membro de um time ágil? *
- ◯ Sim
- ◯ Não

Se sim, por quanto tempo? *
Se não, por favor, insira 0.

Essa experiência como membro do time ágil foi positiva? *
- ◯ Sim
- ◯ Não
- ◯ Não se aplica

Qual(is) outro(s) papel(éis) você desempenhou? *

- [ ] Scrum Master
- [ ] Product Owner
- [ ] Stakeholder
- [ ] Patrocinador
- [ ] Estudante

Por quanto tempo (total) desempenhou esse(s) papel(éis)? *

Tempo em anos e/ou meses. Exemplos: 1 ano e 2 meses, 7 meses, 4 anos.

Em qual(is) contexto(s) essa experiência foi aplicada? *

- [ ] Setor público
- [ ] Iniciativa privada
- [ ] Área Acadêmica

Atualmente, qual sua área de atuação no desenvolvimento de software? *

- ( ) Setor público
- ( ) Iniciativa privada
- ( ) Área Acadêmica

Nos dias de hoje, segue utilizando metodologias ágeis de software? *

- ( ) Sim
- ( ) Não

Qual sua localização (apenas UF)? *

Deseja receber os resultados dessa pesquisa?

Caso queira, informe o seu email no campo abaixo. Caso não queira, apenas desconsidere a pergunta.

Enviar

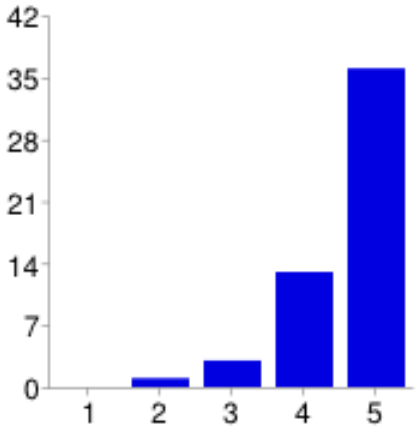
Nunca envie senhas em Formulários Google.

100% concluído.

APÊNDICE VI – RESULTADOS DO QUESTIONÁRIO

Estes foram os resultados e gráficos obtidos para as competências comportamentais:

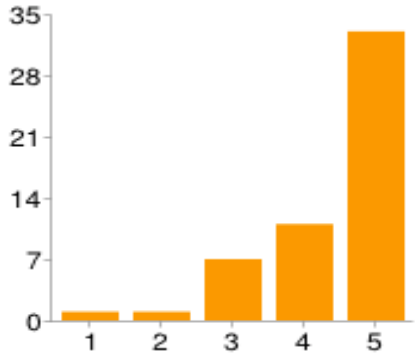
CC 1 – Capacidade de compartilhar e transferir o conhecimento obtido.



1	**0**	0%
2	**1**	2%
3	**3**	6%
4	**13**	25%
5	**36**	68%

Gráfico 8. Resultado da CC1 (O autor).

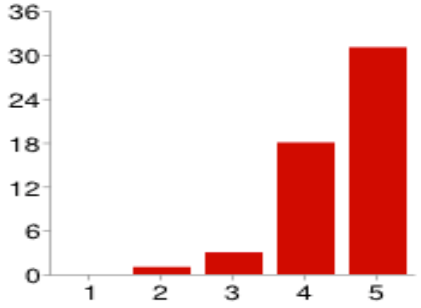
CC2 – Facilidade de comunicação com todos



1	**1**	2%
2	**1**	2%
3	**7**	13%
4	**11**	21%
5	**33**	62%

Gráfico 9. Resultado da CC2 (O autor).

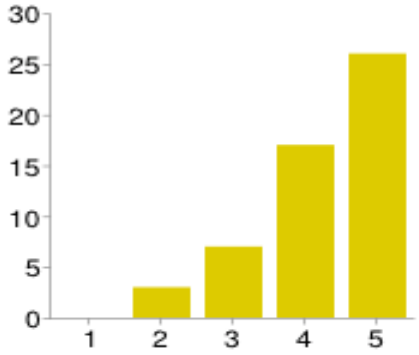
CC3 – Capacidade de trabalhar em equipe e/ou junto com outro membro da equipe



1	**0**	0%
2	**1**	2%
3	**3**	6%
4	**18**	34%
5	**31**	58%

Gráfico 10. Resultado da CC3 (O autor).

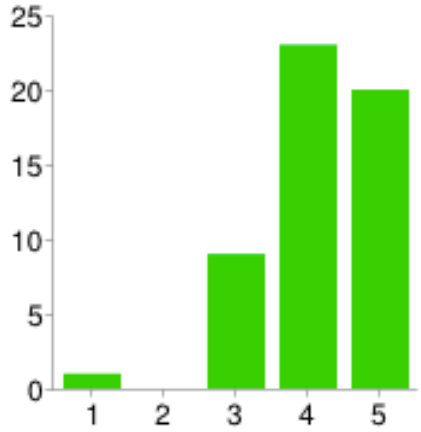
CC4 – Disposição para ajudar o time ou membros do time sempre que for necessário



1	**0**	0%
2	**3**	6%
3	**7**	13%
4	**17**	32%
5	**26**	49%

Gráfica 11. Resultado da CC4 (O autor).

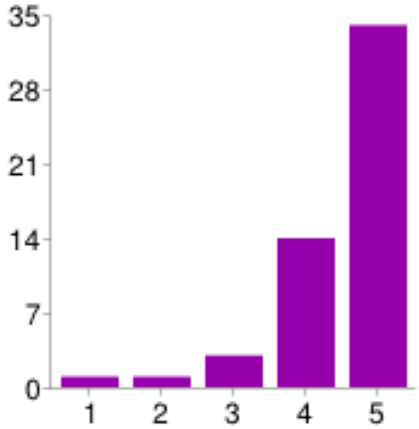
CC5 – Flexibilidade e equilíbrio para gerenciar os conflitos do time



1	**1**	2%
2	**0**	0%
3	**9**	17%
4	**23**	43%
5	**20**	38%

Gráfico 12. Resultado da CC5 (O autor).

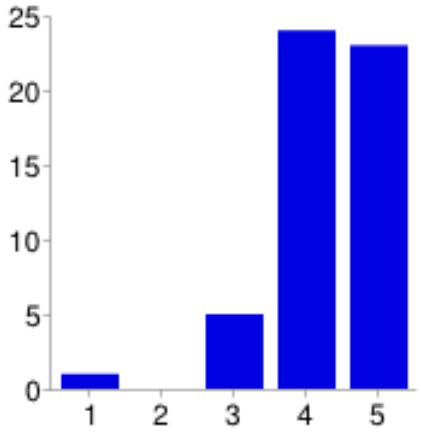
CC6 – Capacidade de cumprir com seus compromissos



1	**1**	2%
2	**1**	2%
3	**3**	6%
4	**14**	26%
5	**34**	64%

Gráfico 13. Resultado da CC6 (O autor).

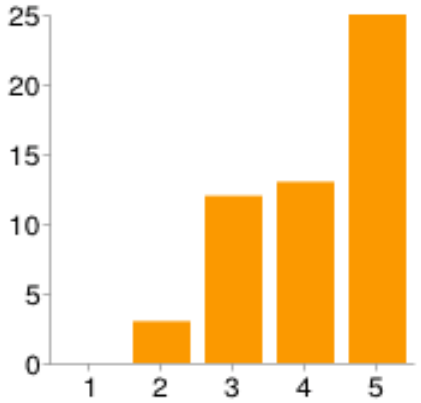
CC7 – Proatividade e iniciativa para a busca de conhecimento e designação de tarefas



1	**1**	2%
2	**0**	0%
3	**5**	9%
4	**24**	45%
5	**23**	43%

Gráfico 14. Resultado da CC7 (O autor).

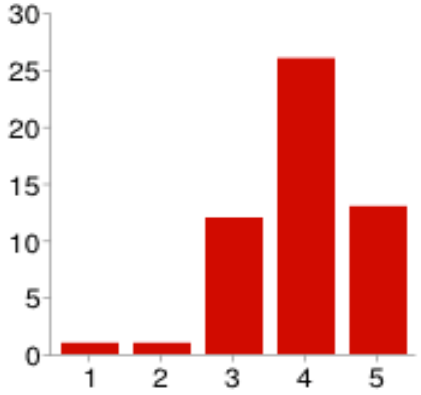
CC8 – Capacidade de estabelecer relacionamentos de confiança com os membros do time



1	**0**	0%
2	**3**	6%
3	**12**	23%
4	**13**	25%
5	**25**	47%

Gráfico 15. Resultado da CC8 (O autor).

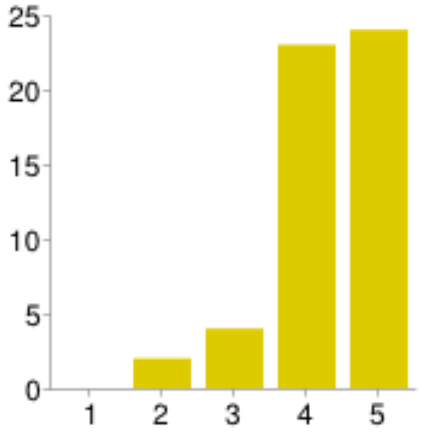
CC9 – Capacidade de tomar decisões diante de situações de incerteza



1	**1**	2%
2	**1**	2%
3	**12**	23%
4	**26**	49%
5	**13**	25%

Gráfico 16. Resultado da CC9 (O autor).

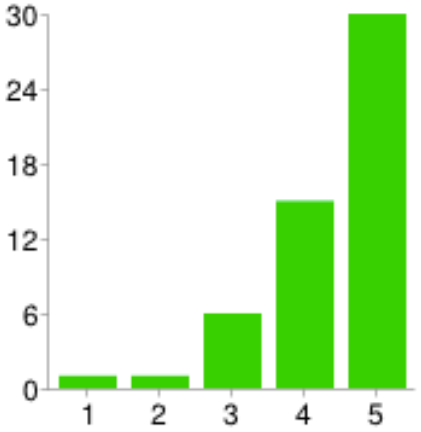
CC10 – Visão focada nos objetivos do time e compartilhada com os membros dele



1	**0**	0%
2	**2**	4%
3	**4**	8%
4	**23**	43%
5	**24**	45%

Gráfico 17. Resultado da CC10 (O autor).

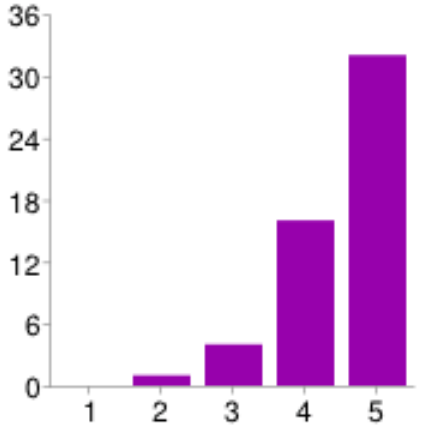
CC11 – Responsabilidade do time por tudo que foi produzido por ele



1	**1**	2%
2	**1**	2%
3	**6**	11%
4	**15**	28%
5	**30**	57%

Gráfico 18. Resultado da CC11 (O autor).

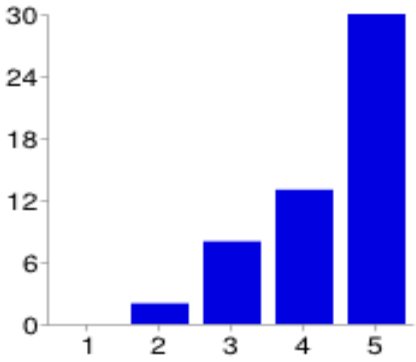
CC12 – Capacidade de evolução e aprendizado com o progresso do trabalho



1	**0**	0%
2	**1**	2%
3	**4**	8%
4	**16**	30%
5	**32**	60%

Gráfico 19. Resultado da CC12 (O autor).

CC13 – Capacidade de auto-gerenciar suas atividades e trabalho

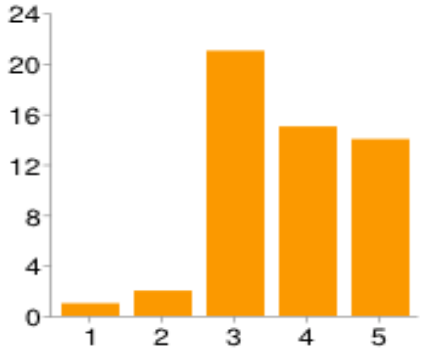


1	**0**	0%
2	**2**	4%
3	**8**	15%
4	**13**	25%
5	**30**	57%

Gráfico 20. Resultado da CC13 (O autor).

Estes foram os resultados e gráficos obtidos para as competências técnicas:

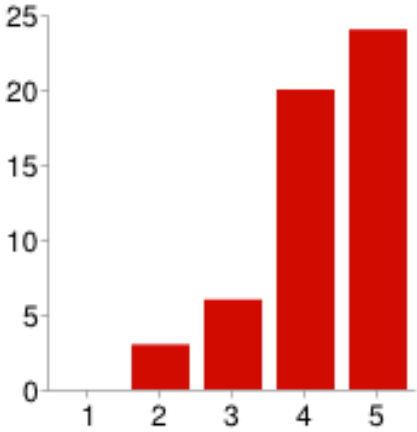
CT1 – Conhecer a instituição em que trabalha para desempenhar seu papel



1	**1**	2%
2	**2**	4%
3	**21**	40%
4	**15**	28%
5	**14**	26%

Gráfico 21. Resultado da CT1 (O autor).

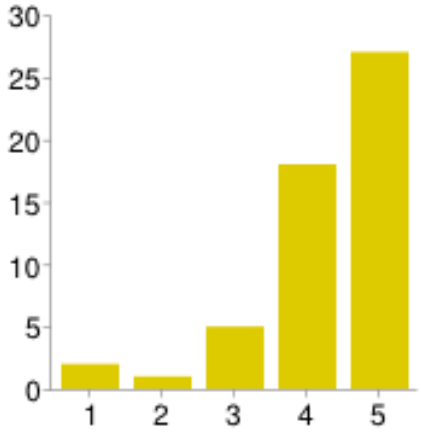
CT2 – Conhecer os valores do desenvolvimento ágil de software



1	**0**	0%
2	**3**	6%
3	**6**	11%
4	**20**	38%
5	**24**	45%

Gráfico 22. Resultado da CT2 (O autor).

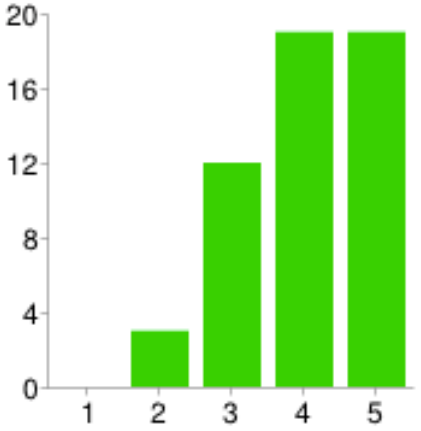
CT3 – Conhecer os princípios do desenvolvimento ágil de software



1	**2**	4%
2	**1**	2%
3	**5**	9%
4	**18**	34%
5	**27**	51%

Gráfico 23. Resultado da CT3 (O autor).

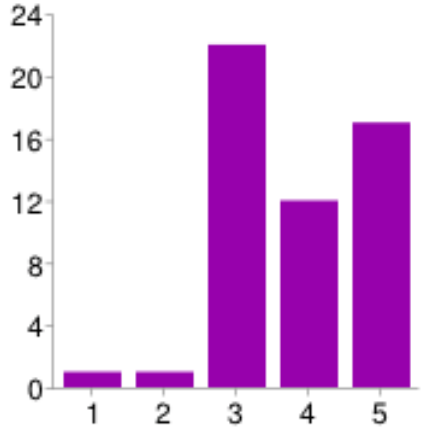
CT4 – Conhecer metodologias ágeis de desenvolvimento de software



1	**0**	0%
2	**3**	6%
3	**12**	23%
4	**19**	36%
5	**19**	36%

Gráfico 24. Resultado da CT4 (O autor).

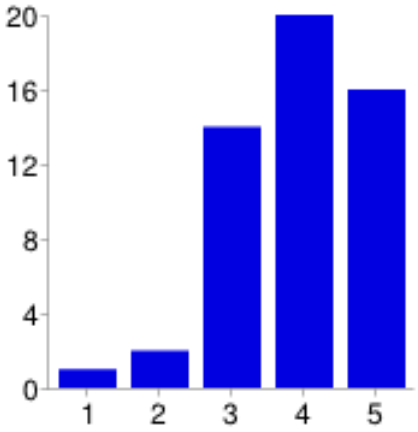
CT5 – Dominar linguagens de programação



1	**1**	2%
2	**1**	2%
3	**22**	42%
4	**12**	23%
5	**17**	32%

Gráfico 25. Resultado da CT5 (O autor).

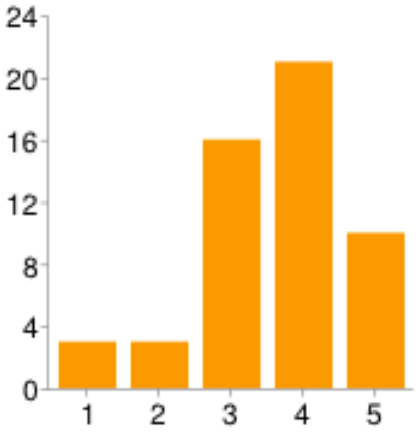
CT6 – Dominar o uso de padrões de codificação



1	**1**	2%
2	**2**	4%
3	**14**	26%
4	**20**	38%
5	**16**	30%

Gráfico 26. Resultado da CT6 (O autor).

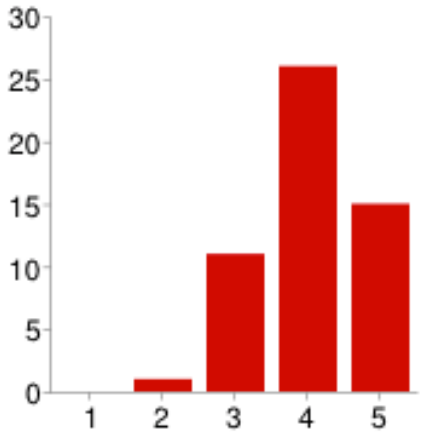
CT7 – Dominar a prática de TDD e seus padrões



1	**3**	6%
2	**3**	6%
3	**16**	30%
4	**21**	40%
5	**10**	19%

Gráfico 27. Resultado da CT7 (O autor).

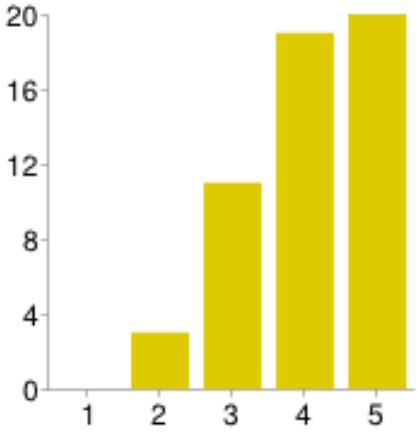
CT8 – Dominar a prática de refatoração de código



1	**0**	0%
2	**1**	2%
3	**11**	21%
4	**26**	49%
5	**15**	28%

Gráfico 28. Resultado da CT8 (O autor).

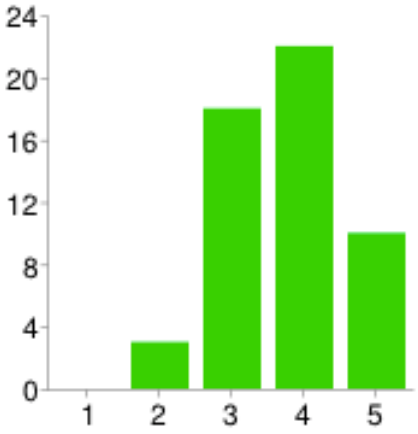
CT9 – Dominar a prática de User stories



1	**0**	0%
2	**3**	6%
3	**11**	21%
4	**19**	36%
5	**20**	38%

Gráfico 29. Resultado da CT9 (O autor).

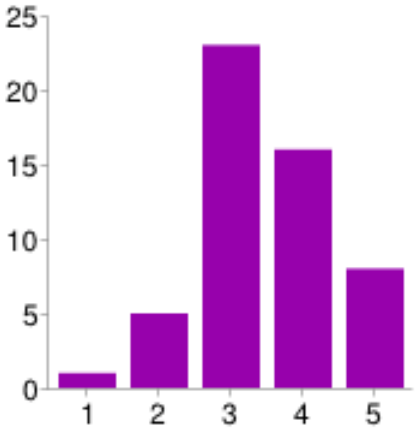
CT10 – Dominar práticas de desenvolvimento da arquitetura do sistema



1	**0**	0%
2	**3**	6%
3	**18**	34%
4	**22**	42%
5	**10**	19%

Gráfico 30. Resultado da CT10 (O autor).

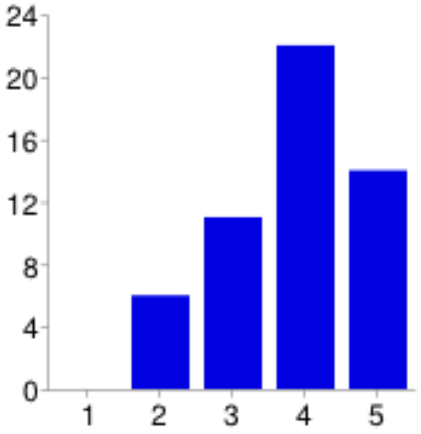
CT11 – Dominar as práticas de desenvolvimento do banco de dados do sistema



1	**1**	2%
2	**5**	9%
3	**23**	43%
4	**16**	30%
5	**8**	15%

Gráfico 31. Resultado da CT11 (O autor).

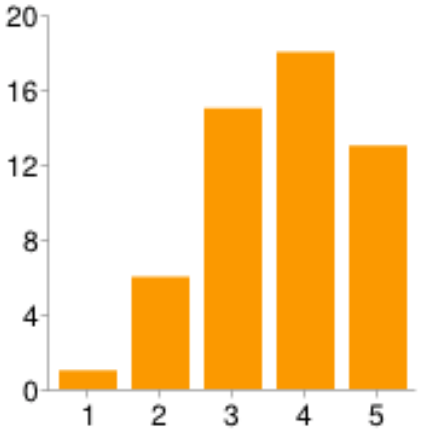
CT12 – Dominar práticas e padrões de gerência de configuração de software



1	**0**	0%
2	**6**	11%
3	**11**	21%
4	**22**	42%
5	**14**	26%

Gráfico 32. Resultado da CT12 (O autor).

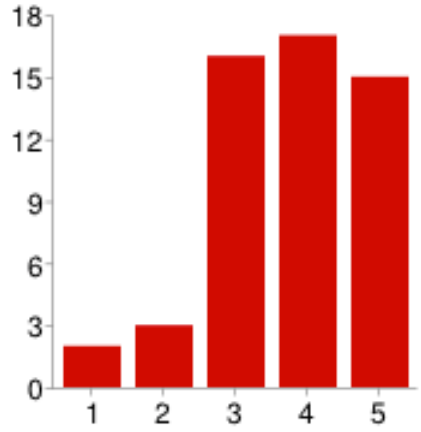
CT13 – Capacidade de construir e configurar o ambiente de desenvolvimento



1	**1**	2%
2	**6**	11%
3	**15**	28%
4	**18**	34%
5	**13**	25%

Gráfico 33. Resultado da CT13 (O autor).

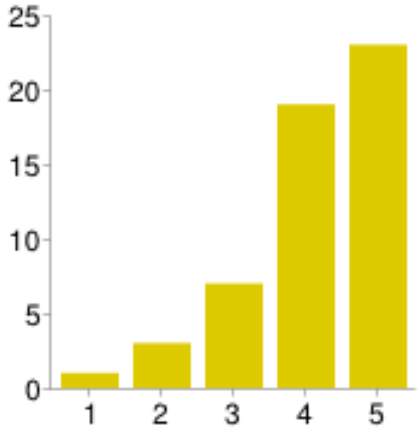
CT14 – Dominar a prática de reuso de software



1	**2**	4%
2	**3**	6%
3	**16**	30%
4	**17**	32%
5	**15**	28%

Gráfico 34. Resultado da CT14 (O autor).

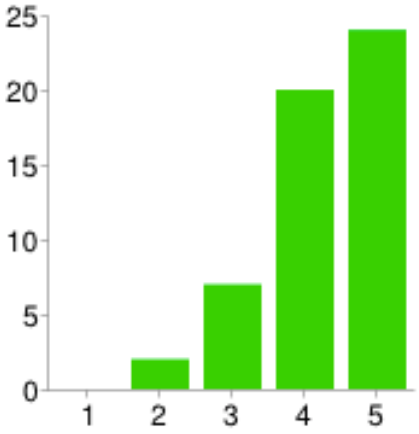
CT15 – Capacidade de negociar o escopo do trabalho



1	**1**	2%
2	**3**	6%
3	**7**	13%
4	**19**	36%
5	**23**	43%

Gráfico 35. Resultado da CT15 (O autor).

CT16 – Capacidade de estimar esforço de acordo com suas limitações



1	**0**	0%
2	**2**	4%
3	**7**	13%
4	**20**	38%
5	**24**	45%

Gráfico 35. Resultado da CT16 (O autor).