INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS

LARISSA DINIZ NAZARETH RESENDE

APLICAÇÃO DO *SCRUM* E DE PRINCÍPIOS ÁGEIS NA GESTÃO DE PROJETOS SOLO DE *SOFTWARES*

CONGONHAS
2024

RESUMO

Trata-se de um estudo que visa analisar as metodologias ágeis, com enfoque no *Scrum*, sendo essa ferramenta uma alternativa para as metodologias tradicionais de gerenciamento de projetos. Sabe-se que atualmente é possível usar metodologias e também princípios ágeis em projetos de *softwares*, inclusive nos projetos com apenas um desenvolvedor. O presente trabalho teve como objetivo analisar as aplicações do *Scrum* e de princípios ágeis na gestão de projetos solo de *softwares.* Foi realizado um estudo de caso descritivo, a partir da aplicação de entrevistas semiestruturadas com seis profissionais da área de desenvolvimento de *software*. A partir das entrevistas foi possível criar um quadro com os resultados, que consistiu em levantar aspectos específicos da aplicação do *Scrum* em projetos solo a partir de categorias temáticas predefinidas, obtendo como resultado a conclusão de que é viável a adaptação do *Scrum* para projetos solo. Entretanto, foi constatada e destacada a relevância da experiência do desenvolvedor, que deve ter habilidades interpessoais e uma comunicação assertiva com o cliente, além de outros pontos que foram analisados e discutidos neste trabalho.

Palavras- chave: desenvolvimento solo, *softwares*, metodologias ágeis, *Scrum.*

ABSTRACT

This study aims to analyze agile methodologies, focusing on Scrum, with this tool serving as an alternative to traditional project management methodologies. It is known that nowadays it is possible to use methodologies and agile principles in software projects, including projects with only one developer. The present work aimed to analyze the applications of Scrum and agile principles in the management of solo software projects. A descriptive case study was conducted, based on the application of semi-structured interviews with six professionals in the software development field. From the interviews, it was possible to create a table with the results, which consisted of identifying specific aspects of Scrum application in solo projects based on predefined thematic categories, resulting in the conclusion that it is feasible to adapt Scrum for solo projects. However, the relevance of the developer's experience was found and highlighted, as they must have interpersonal skills and assertive communication with the client, in addition to other points that were analyzed and discussed in this work.

Keywords: *solo development*, *software*, *agile methodologies*, *Scrum*.

1. INTRODUÇÃO

O *software* (programa de computador), é um conjunto de instruções e dados que permitem a um computador realizar tarefas específicas. A necessidade da utilização de *softwares* no mundo atual reflete as transformações ocorridas na sociedade devido à digitalização e a facilidade de acesso a tecnologias. Inicialmente, *softwares* eram ferramentas restritas a grandes empresas e instituições acadêmicas, focadas em tarefas específicas como cálculos complexos e processamento de dados em grande escala. Contudo, com o avanço tecnológico, especialmente a disseminação da internet e a popularização de dispositivos pessoais inteligentes, o *software* tornou-se uma parte intrínseca da vida cotidiana. Devido a isso e ao fato de cada vez mais os programas necessitarem de mais recursos, a produção do *software* como produto se tornou mais complexa, o que traz desafios ao seu processo de desenvolvimento (SOMMERVILLE, 2011).

Com base nisso, Pressman (2006) explica que o processo de desenvolvimento de *softwares* abrange uma série de etapas consecutivas, que incluem a concepção inicial, a análise de requisitos, o *design*, a implementação (codificação), o teste, a implantação e a manutenção do *software*. O desafio principal é produzir um produto que atenda às necessidades dos usuários finais enquanto se mantém dentro de um cronograma e orçamento previstos. Esse processo pode adotar várias metodologias, como o modelo em cascata, que é o mais tradicional, clássico, sequencial e rigoroso, ou abordagens ágeis, que são iterativas e adaptativas, permitindo flexibilidade e entrega contínua. A escolha da metodologia depende das especificidades do projeto, incluindo seus objetivos, complexidade e os envolvidos no processo de desenvolvimento. Geralmente as metodologias mais ágeis se adaptam melhor ao desenvolvimento de *softwares* mais atuais, que são mais complexos e com mais funcionalidades a serem desenvolvidas (PRESSMAN, 2006).

Segundo Sbrocco e Macedo (2012), o desenvolvimento de *software* é uma atividade complexa e pode tornar-se caótica se não forem utilizadas as metodologias e técnicas adequadas. Segundo Sutherland (2014), entre 50% e 85% de todo *software* desenvolvido é desperdiçado, justamente devido a utilização de modelos antigos e ultrapassados de gerenciamento de projetos que ainda vem sendo utilizado pelas empresas.

Assim, utilizar métodos de desenvolvimento de *software* considerados ultrapassados para criar *softwares* modernos pode acarretar uma série de problemas que afetam a eficiência e a qualidade do produto final (PRESSMAN, 2006). Métodos rígidos e sequenciais, como o modelo em cascata, podem não se adaptar bem às necessidades dinâmicas e aos rápidos ciclos

de inovação exigidos pelo mercado atual. A incapacidade de responder prontamente a mudanças nos requisitos ou adaptar-se a novas tecnologias pode resultar em um produto obsoleto antes mesmo de seu lançamento. Além disso, a falta de flexibilidade nos processos de desenvolvimento pode aumentar os custos, prolongar os cronogramas de entrega e reduzir a satisfação do usuário final (PRESSMAN, 2006).

Outro desafio com relação ao gerenciamento de projetos de desenvolvimento de *softwares* está em compreender os problemas e propor soluções que atendam as expectativas dos clientes. Não são raros os projetos que ao final não conseguem resolver o real problema, ou seja, o sistema desenvolvido não gera o valor para o qual foi criado (SILVA; LOVATO, 2016). Visando solucionar essa lacuna no desenvolvimento de projetos, o modelo em cascata, marcado por fases, já não é o mais indicado para o desenvolvimento de grande parte dos projetos atuais, que exigem adaptabilidade e mudanças rápidas, surgindo como alternativa os métodos ágeis (MENDOZA; PIZZOLATO, 2023). Nesse modelo tradicional, cada fase deve ser finalizada para que aconteçam avanços no desenvolvimento do projeto, fazendo com que as etapas do modelo sigam de forma linear. Uma das principais críticas a essa metodologia é a dificuldade de controle, pois a cada alteração em determinado ponto do projeto, será necessário voltar ao início para alteração ou documentação (PRESSMAN, 2006).

Em contrapartida ao exposto, em 2001, um grupo de especialistas criou o manifesto ágil, que consiste em uma mudança de valores e princípios para nortear o processo ágil de criação de *softwares* (BECK *et al.*, 2001). Ao contrário das metodologias tradicionais, as metodologias ágeis não precisam necessariamente obedecer ao modelo sequencial, sendo uma abordagem mais dinâmica e flexível (SOMMERVILLE, 2011).

Normalmente, o desenvolvimento dos projetos e o uso das metodologias ágeis são realizados por equipes, uma vez que projetos complexos demandam diversas habilidades e especializações. Contudo, muitos desenvolvedores optam por trabalhar sozinhos, seja por preferência pessoal, pela natureza do projeto ou por restrições de recursos (PREVIATO, 2018). Dessa maneira, no desenvolvimento solo o uso de uma metodologia ou a combinação de metodologias pode ser uma alternativa viável, mesmo que a metodologia tenha sido pensada e desenvolvida para o uso coletivo (PREVIATO, 2018). Desse modo, já é possível encontrar pesquisas com o objetivo de adaptar as metodologias ágeis para serem aplicadas em desenvolvimentos solo, que serão citadas no decorrer desse trabalho

Dentre as diversas metodologias ágeis, o *Scrum* é um dos modelos mais utilizados atualmente. Trata-se de um método que visa entregar resultados de maneira incremental, ou seja, o produto é construído e entregue em partes sucessivas utilizáveis, permitindo revisões e

ajustes baseados nos retornos dos usuários após cada parte desenvolvida, além disso, busca resultados efetivos e com baixo custo (SOMMERVILLE, 2011).

Desta forma, este trabalho teve como objetivo principal analisar as aplicações do *Scrum* e de princípios ágeis na gestão de projetos solo de *softwares* e como objetivos específicos: a) efetuar uma pesquisa sobre métodos ágeis e suas possíveis aplicações no desenvolvimento de *softwares*; b) analisar pesquisas existentes que buscaram a aplicação de metodologias e princípios ágeis em projetos solo de *softwares*; c) analisar a possibilidade do uso do *Scrum* como metodologia a ser aplicada no desenvolvimento solo de *softwares*; d) fazer uma entrevista com profissionais da área para obter *feedbacks* e validar as informações obtidas ao longo do processo de pesquisa; e) analisar as informações obtidas na entrevista com os profissionais para gerar os resultados, concluindo sobre a viabilidade da aplicação do *Scrum* para projetos solo de desenvolvimento de *software*.

O presente artigo foi estruturado da seguinte forma: no tópico 2 foram realizados o referencial teórico, que foi dividido em três partes. No tópico 3 foram abordados os procedimentos metodológicos utilizados. O tópico 4 consistiu na apresentação dos resultados obtidos por meio das entrevistas e no tópico 5 foi realizada a discussão dos resultados, considerando as entrevistas e as literaturas sobre o tema. Por fim, no tópico 6 foi realizada a conclusão, com as devidas considerações finais pertinentes ao trabalho.

2. REFERENCIAL TEÓRICO

O referencial teórico foi dividido em três partes. Inicialmente, foi feita uma contextualização sobre metodologias ágeis. Na segunda parte, tratou-se especificamente do *framework Scrum* e na terceira parte foi descrita especificamente a aplicação do *Scrum* em projetos solo.

2.1 METODOLOGIAS ÁGEIS

Novos métodos vêm surgindo para proporcionar maior agilidade na execução de projetos. Esses métodos são conhecidos como métodos ágeis e são focados na qualidade e agilidade na gestão de projetos. A agilidade pode ser definida como a capacidade de responder rapidamente a mudanças em um ambiente incerto nos negócios. Ela está relacionada a capacidade de oferecer produtos ou serviços aos clientes com um mesmo resultado, mas em ciclos de desenvolvimento mais reduzidos (XU; KOIVUMÄKI, 2019). Como exemplos de

métodos ágeis, pode-se citar o *Scrum, Kanban, eXtreme Programming (XP), Feature Driven Development* (FDD) e o *Lean* (SILVA; SOUZA NETO, 2015).

As metodologias ágeis representaram um marco para a gestão de projetos, visto que auxiliam equipes a atingir de forma sistemática a disciplina necessária para a execução e a inovação contínua, algo que era difícil de se conseguir com a burocracia hierárquica relatada nas metodologias tradicionais (DENNING, 2013). Os métodos ágeis se caracterizam por serem incrementais, cooperativos, diretos e adaptativos. Podem ser considerados incrementais, pois as entregas são diversas durante o andamento do projeto, além de possuírem melhorias e rápidos ciclos de desenvolvimento. São cooperativos, por causa das interações entre o cliente e a equipe do projeto. Além disso, são *diretos*, pois o método em si é de fácil aprendizado e contém apenas a documentação estritamente necessária. Por fim, são adaptativos, devido à habilidade de realizar mudanças ágeis no decorrer do projeto (ABRAHAMSSON *et al.*, 2003).

No contexto das metodologias ágeis, a gestão ágil de projetos surge como resposta à crescente necessidade de inovação e urgência no desenvolvimento de projetos. Porém, não enfatiza apenas a necessidade de construir produtos rapidamente e de forma adaptável, mas também a importância de desenvolver equipes com as mesmas características (HIGHSMITH, 2004). Assim, conforme Sommerville (2011), as habilidades da equipe de desenvolvimento devem ser reconhecidas e exploradas, considerando que membros das equipes devem desenvolver suas próprias maneiras de trabalhar sem processos prescritos, ou seja, devem ser equipes auto organizadas.

2.2 *SCRUM*

De acordo com Schwaber e Sutherland (2016, p.3), a metodologia *Scrum* baseia-se em um "*framework* leve que ajuda pessoas, times e organizações a gerar valor por meio de soluções adaptativas para problemas complexos". Em condições normais, a utilização do *Scrum* para o desenvolvimento de *softwares* ocorre da seguinte forma: primeiramente o representante do cliente (*Product Owner*) define suas necessidades em uma lista denominada *Product Backlog*, que trata-se de uma lista de funcionalidades do sistema a ser desenvolvido com todas as suas características, melhorias, requisitos, correções e regras. Os itens dessa lista são ordenados por prioridade e apresentados à equipe de desenvolvimento (*Development Team*), que avalia o tempo necessário para atendê-los (SILVA, 2011).

Realiza-se então a reunião de Planejamento da *Sprint* (*Sprint Planning*), para estabelecer quais funcionalidades serão implementadas em um período predeterminado, chamado de *Sprint* (SILVA, 2011). De acordo com Bastos e Bastos (2021), os *Sprints* podem ser considerados

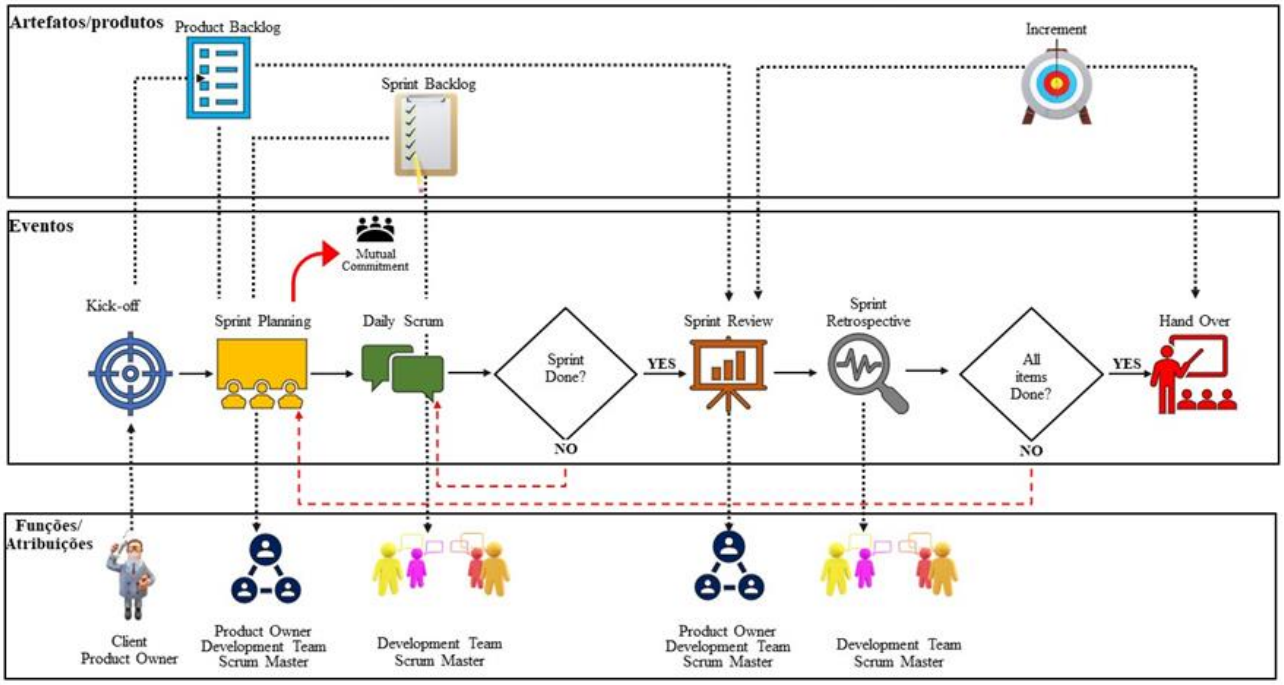
ciclos de execução do projeto, que ocorrem com uma duração que varia entre uma semana e um mês. De acordo com Schwaber e Sutherland (2016), o *Sprint planning* permite aumentar a previsibilidade da *Sprint*, garantindo formas de inspeção e adaptação do progresso pelo menos uma vez por mês. Assim, durante a execução da *Sprint*, os integrantes do grupo informam diariamente o estado de suas atividades por meio de reuniões rápidas (*Daily Scrum*), viabilizadas pelo *Scrum Master*, responsável por ajudar o time a adotar o *framework* e eliminar possíveis impedimentos.

Ao final de cada ciclo é realizada a revisão da *Sprint* (*Sprint Review*), na qual a nova versão do sistema é apresentada e avaliada, assim como as próximas tarefas são escolhidas. Uma reunião retrospectiva da *Sprint* (*Sprint Retrospective*) é então realizada, de forma a revisar as contribuições e os problemas encontrados, buscando por soluções adequadas. Nessa etapa a *Sprint* é avaliada sob a perspectiva do processo, equipe e produto (SILVA, 2011).

Além do *Product Backlog* o *Scrum* faz o uso de mais dois artefatos para garantir a transparência das informações aos envolvidos: o *Sprint Backlog* e o Incremento. O *Sprint Backlog* é uma lista que contém parte das funcionalidades do *Product Backlog*, as quais são destacadas para serem desenvolvidas em cada *Sprint*. Já o Incremento, consiste no resultado da *Sprint* como uma versão potencialmente utilizável, formada pelo resultado atual da *Sprint*, somado aos resultados das *Sprints* anteriores (SILVA, 2011).

Segundo Rising e Janoff (2000), a implementação do *Scrum* é baseada em evoluções, melhorias e aprendizados contínuos, uma vez que em cada interação sejam detectadas falhas e possíveis melhorias do processo a serem aplicadas na próxima interação. Martins *et al.* (2009) reforçam que essas características contribuíram para que a metodologia *Scrum* obtivesse um enorme ganho no setor de produção de *software*, por conta de ser uma metodologia que se adapta com facilidade, com grande praticidade e principalmente onde é possível realizar alterações e mudanças nos requisitos e funcionalidades durante o processo de desenvolvimento. Na Figura 1, pode-se visualizar o processo *Scrum* descrito neste tópico.

Figura 1 - Processo Scrum e seus elementos



Fonte: Adaptado de Poudel *et al.* (2020).

2.3 PROJETOS SOLO DE SOFTWARE E APLICAÇÃO DO *SCRUM*

A elaboração de um *software* consiste em um projeto com todas as suas características eminentes. Um projeto é um empreendimento progressivo, temporário e único, com o objetivo de criar um produto, serviço ou mesmo gerar um resultado exclusivo. É progressivo, pois sua elaboração é fundamentada em incrementos interativos ao longo de seu ciclo de vida; temporário, uma vez que tem início e término bem definidos e único pelo fato de produzir algo diferente de outros produtos ou serviços anteriores (PMI, 2013).

No contexto de desenvolvimento de projetos de *softwares*, Fadel e Silveira (2010) defendem que aderir a uma metodologia ágil permite que desenvolvedores atinjam as expectativas esperadas por seus usuários e sejam capazes de fazer as mudanças necessárias no decorrer do projeto. No caso de desenvolvedores solo, esses vão desenvolver e ao mesmo tempo gerenciar os projetos, ou seja, precisam encontrar uma metodologia que os ajude a aumentar a produtividade, diminuir os gastos, o tempo de trabalho e que possibilite alterações sem perder a qualidade do produto final.

Algumas pesquisas já foram realizadas com o intuito de analisar o uso de metodologias ágeis em projetos de desenvolvimento solo de *softwares*, ou seja, quando se tem uma única pessoa responsável pela execução e gerenciamento do projeto. Pagotto *et al.* (2016) desenvolveu e aplicou uma metodologia, que se caracteriza como um processo iterativo e

incremental que une as boas práticas delineadas pelo *Personal Software Process (PSP)* e o *Scrum*. O *Scrum* já foi anteriormente detalhado e o PSP consiste em um processo de melhoria projetado para ajudar os desenvolvedores a controlar, administrar e aperfeiçoar sua competência para produzir *software* de qualidade, baseando-se no princípio do conhecimento, avaliação e melhorias contínuas do processo individual. A metodologia usada Pagotto *et al.* (2016) conta com artefatos e atores que atuam em um processo dividido em quatro atividades: a) *Requirement*, pela qual surgem três artefatos, sendo eles o escopo, o *Product Backlog* e o protótipo do *software*; b) *Sprint*, que conta com quatro artefatos, compreendendo o *Sprint Backlog*, o produto, a ata e a planta de desenvolvimento; c) *Deployment*, que se utiliza de dois artefatos (produto e ata de validação), com o objetivo de colocar o *software* em fase de teste, permitindo com que o cliente possa avaliar o produto desenvolvido; e d) *Management*, composto por quatro artefatos, sendo eles a Estrutura Analítica do Projeto (EAP), planilhas de custo, planilha de controle e o cronograma, sendo o objetivo dessa etapa o de monitorar o projeto.

Além dessas atividades, Pagotto *et al*. (2016) define quatro atores principais: a) o *product owner*, é aquele que conhece o produto que será desenvolvido e tem a responsabilidade de fornecer as informações necessárias para o seu desenvolvimento; b) o desenvolvedor individual, que diferentemente da metodologia *Scrum* em que há uma equipe, é o único responsável pelo desenvolvimento e gerenciamento; c) o orientador, que corresponde àquela pessoa que conhece o processo, a tecnologia, o escopo, podendo atuar como um consultor durante a elaboração do *software*; d) o grupo de validação, responsável por atestar o produto/*software* desenvolvido. O autor desse método testa a aplicação dessa metodologia por vários alunos da Universidade Federal do Paraná dos cursos de Engenharia da Computação e Tecnologia em Análise e Desenvolvimento de Sistemas da UTFPR - *campus* Cornélio Procópio entre os anos de 2012 e 2014. Esses alunos relataram que o processo apresentado supre as necessidades eminentes na produção de *software*s atendendo às expectativas inerentes a gestão de projetos do cliente e do desenvolvedor (PAGOTTO *et al*., 2016).

Outra proposta de metodologia para desenvolvimento de *softwares* solo é proposta por Previato (2018), que utiliza como base a metodologia ágil *Scrum*, fazendo uma adaptação da metodologia. Logo, muitos conceitos, artefatos, eventos e funções são aplicadas sem alteração. A proposta é dividida em três partes: definição, construção e implantação, sendo que essa adaptação pode ser usada em instituições ou empresas que demandam automatizações de processos ou qualquer outro serviço que possa ser informatizado por um único indivíduo.

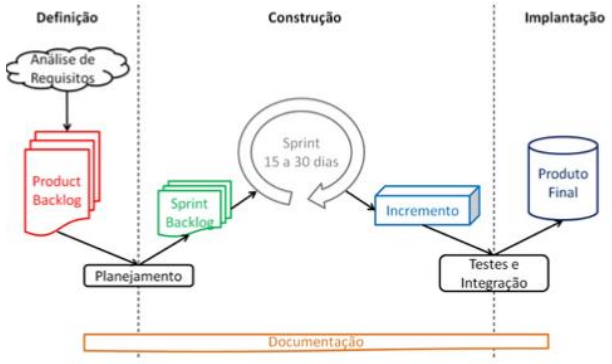
Na parte de definição Previato (2018) sugere que uma ou mais reuniões sejam realizadas juntamente com os solicitantes. Nessa reunião serão definidas as funcionalidades da aplicação e uma lista com as funcionalidades que serão priorizadas, sendo essa etapa nomeada como Análise de Requisitos. Após é feito o *Product Backlog*, uma lista de funcionalidades do sistema a ser desenvolvido com todas as suas características, melhorias, requisitos, correções e regras.

Na etapa de construção será feita as *Sprints Planning*, que consistirá em um planejamento ao início de cada *Sprint* para determinar quais funcionalidades serão implementadas, esse planejamento será feito com o solicitante ou apenas pelo desenvolvedor. Será feito também a *Sprint Backlog* com as tarefas que deverão ser desenvolvidas em cada *Sprint e*, ao final de cada uma, uma ou mais funcionalidades poderão ser testadas pelos solicitantes, cujo resultado é nomeado como Incremento, assim como na proposta original do *Scrum*, e poderá ser testado pelo cliente (PREVIATO, 2018).

A reunião diária (*Daily Scrum)* não é realizada como descrito na literatura existente sobre o *Scrum*, ou seja, no desenvolvimento solo é necessário fazer uma adaptação, o que não impede o desenvolvedor de separar um tempo diário para verificar o que foi realizado, se existem impedimentos e se o que foi feito ajudou a atingir a meta da *Sprint*. A revisão da *Sprint* (*Sprint Review*) também é realizada pelo desenvolvedor para verificar o que deu certo e o que deu errado.

Por fim, na etapa de implantação, na proposta para desenvolvimento solo devem ser realizados testes, implementando as novas funções, que serão feitas pelo solicitante antes da entrega definitiva, validando assim todo o *Product Backlog*. É importante ressaltar que a documentação deve ser a menor possível, porém suficiente para que se tenha registro do que é a aplicação (PREVIATO, 2018). Na Figura 2 pode-se ter uma visão geral da proposta de aplicação do *Scrum* em projetos solo.

Figura 2 - Visão geral da proposta de aplicação do Scrum em projetos solo.



Fonte: Previato (2018)

Essa metodologia proposta foi aplicada pela equipe de desenvolvimento da Seção Técnica de Informática do ICMC-USP em três projetos pilotos, tendo obtido bons resultados nos três projetos com a aplicação de princípios ágeis e a adaptação da metodologia *Scrum* (PREVIATO, 2018). No Quadro 1 foi feita uma comparação entre o *Scrum*, o *Scrum* Solo (Pagotto *et al.*, 2016) e a adaptação do *Scrum* para projetos de desenvolvimento solo proposta por Previato (2018).

Quadro 1 – Comparação entre as metodologias

Elementos	Atividades	Artefatos/Produtos	Eventos	Funções/Atribuições
Scrum	-	1. *Product Backlog* 2. *Sprint Backlog* 3. *Incremento*	1. *Sprint Planning* 2. *Daily Scrum* 3. *Sprint Review* 4. *Sprint Retrospective*	1. *Product Owner* 2. *Development Team* 3. *Scrum Master*
***Scrum* Solo (Pagotto *et al.* (2016)**	***Requirement***	1. Escopo 2. *Product Backlog* 3. Protótipo do *software*	1. Reuniões que envolvem o *Product Owner*, o desenvolvedor e o orientador.	1. Product Owner. 2. Desenvolvedor individual 3. Orientador 4. Grupo de validação
	Sprint	1. *Sprint Backlog* 2. Produto ou parte dele em funcionamento 3. Ata 4. Planta de Desenvolvimento		
	Deployment	1. Produto 2. Ata de validação		
	Management	1. Estrutura analítica do projeto (EAP) 2. Planilha de custo 3. Planilha de controle 4. Cronograma		
Adaptação do *Scrum* para projetos solo Previato (2018)	**Definição**	1. *Product Backlog*	1. Análise de Requisitos	1. Pessoas interessadas (solicitante) 2. Desenvolvedor individual
	Construção	1. *Sprint Backlog* 2. *Incremento*	1. *Sprints Planning (planejamento feito pelo desenvolvedor)*	
	Implantação	1. Produto Final	1. Testes e Integração	

Fonte: Elaborado pela autora

3. METODOLOGIA

Este artigo, teve como objetivo analisar a aplicação do *Scrum* e de princípios ágeis para a gestão de projetos solo de *softwares*. Para fazer essa avaliação buscou-se fazer um estudo de caso com a participação de seis profissionais da área da computação e engenharias, que possuem experiência no desenvolvimento e gerenciamento desse tipo de projeto. O estudo foi baseado a partir da ótica desses indivíduos, profissionais que trabalham ou trabalharam em empresas, para que fosse analisada a viabilidade de aplicação também no setor privado visto que as duas pesquisas citadas nesse artigo foram aplicadas e testadas de forma majoritária por indivíduos e profissionais inseridos no contexto acadêmico.

O estudo de caso é um método específico de pesquisa de campo que visa analisar em detalhes um fenômeno exatamente como ocorreu, sem intervenção significativa do pesquisador (GIL, 2008). Tem o objetivo de aprofundar ao máximo no objeto analisado, que nesse trabalho foi a análise da aplicação do *Scrum* e de princípios ágeis em desenvolvimentos solo de *softwares*, analisando a viabilidade, vantagens, desvantagens e fatores de influência. Considerando o objetivo do artigo, o estudo de caso realizado pode ser classificado como descritivo. As pesquisas de caráter descritivo buscam discorrer sobre a relação entre variáveis e expor as suas principais características, além disso, buscam investigar de forma prática como os fenômenos se comportam (MARCONI; LAKATOS, 2003). As pesquisas descritivas são feitas por meio de relatos estruturados, sendo direcionado à interpretação dos fatos sem a interferência do pesquisador (PRODANOV; FREITAS, 2013).

Esse trabalho usou uma abordagem qualitativa. Segundo Sampieri (2006), o enfoque qualitativo dá profundidade aos dados, a dispersão, a riqueza interpretativa, a contextualização do ambiente, os detalhes e as experiências únicas. Também oferece um ponto de vista recente, natural e holístico dos fenômenos, assim como flexibilidade, o que atende ao objetivo proposto neste estudo, visto que foram consideradas as experiências únicas e individuais dos seis profissionais que participaram do estudo. A pesquisa de abordagem qualitativa proporciona um leque de métodos de investigação e coletas de dados, que podem ser através de entrevistas, observações, documentos e registros, sendo escolhida para esse trabalho como forma de obtenção de dados a entrevista, por acreditar ser através deste método a aquisição de resultados mais fidedignos com relação aos objetivos aqui desejados.

A coleta de dados ocorreu por meio de entrevistas semiestruturadas com os seis profissionais. Conforme Laville e Dionne (1999), a entrevista semiestruturada proporciona uma maior flexibilidade à coleta de dados, assim como uma maior abertura ao entrevistado, tornando

as respostas mais fiéis, pois são feitas verbalmente em uma ordem prevista, mas, na qual o entrevistador pode acrescentar perguntas de esclarecimento. As entrevistas foram baseadas em um roteiro previamente estabelecido, o qual foi dividido em 9 categorias: informações pessoais, percepção sobre o *Scrum*, membros do projeto, *Scrum Master* e *Product Owner*, *Product Backlog*, *Sprint*, tempo e agilidade, satisfação do cliente e aspectos gerais. Para o tratamento dos dados, optou-se por seguir as mesmas categorias temáticas utilizadas para a entrevista. "Para isso, o investigador buscou encontrar as categorias que são expressões ou palavras significativas em função das quais o conteúdo de uma fala foi organizado" (MINAYO, 2006, p.317).

Foram então realizados recortes de trechos relevantes das entrevistas, os quais foram comparados e analisados dentro de cada categoria. As categorias foram interpretadas e discutidas com base no referencial teórico.

4. APRESENTAÇÃO DO RESULTADO

Neste tópico foram apresentados os resultados obtidos na pesquisa, que foram ordenados de acordo com as categorias temáticas escolhidas, sendo elas: percepção sobre o *Scrum*, membros do projeto, *Scrum Master* e *Product Owner*, *Product Backlog*, *Sprint*, tempo e agilidade, satisfação do cliente e aspectos gerais. As entrevistas foram conduzidas com a participação de seis profissionais, sendo três analistas de sistemas de informação, dois engenheiros da computação e um cientista da computação, que aconteceram de forma individual.

4.1 PERCEPÇÃO SOBRE O *SCRUM*

Todos os profissionais acreditam que a ferramenta *Scrum* é útil no desenvolvimento dos projetos que realizam. Os dois pontos principais levantados pela maioria dos entrevistados foi que a metodologia é útil na organização do projeto e também auxilia nas alterações, visto que outras metodologias mais tradicionais valorizam o processo de documentação. De acordo com um dos entrevistados pode-se perceber que uma das grandes vantagens do uso dessa metodologia é a sua flexibilidade:

> "O *Scrum* é extremamente útil, pois ele permite uma maior facilidade para executar alterações, outras metodologias mais tradicionais focam muito em documentações e acabam "travando" mais as mudanças no projeto. Além disso, o fato de o desenvolvimento ser dividido em *Sprints* curtas, faz com que "rapidamente" tenhamos soluções utilizáveis, bem como já permite aos clientes testarem essas funcionalidades desenvolvidas e já fornecerem um *feedback*, caso algo não esteja bom, o custo de alteração é menor, pois a mudança ocorre em uma parcela menor do projeto."

Ainda nessa categoria temática foi perguntado sobre a viabilidade de aplicação em desenvolvimentos solo de *softwares*. Com base na experiência e conhecimento de cada entrevistado, quatro deles acreditam que a metodologia pode ser usada e adaptada para projetos solo e dois dos entrevistados não acreditam que seja viável e eficiente a adaptação desta metodologia aos projetos solo. Os quatro profissionais que acreditam na viabilidade relataram que pequenas adaptações já permitiriam o uso da ferramenta e também que alguns papéis seriam eliminados ou ajustados para que ficassem apenas sob responsabilidade do desenvolvedor. Um dos entrevistados explicou que nesse tipo de projeto, por exemplo, as reuniões diárias (*Daily Scrum)* não seriam necessárias, mas que poderiam facilmente ser substituídas por uma análise diária feita pelo próprio desenvolvedor.

Os dois entrevistados que não acreditam na possibilidade de aplicação da metodologia, levantaram questões como a falta de conhecimento da metodologia, complexidade da metodologia e também acreditam que existem metodologias melhores e mais aplicáveis a projeto solo, como a XP (*eXtreme Programming)* e outros modelos evolucionários. De acordo com um dos entrevistados que não acredita na possibilidade de adaptação:

> "[...] a falta de conhecimento da metodologia faz com que o desenvolvedor não adapte a metodologia para o desenvolvimento solo. Também acredito que existam outras metodologias melhores dependendo do caso. Dependendo do projeto, pode-se utilizar o Kanban dividindo as atividades no *Trello*, por exemplo."

Um dos entrevistados acredita que o *Scrum* só faz sentido quando utilizado em equipe, tornando a adaptação a projetos solo relativamente complexa. Os quatro profissionais que acreditam na viabilidade de adaptação acreditam que as *Sprint Planning, as Daily Scrum, as Sprint Review* e as *Sprint Retrospective* são os principais eventos que sofreriam mudanças, pois não poderiam ser aplicados de acordo com a literatura disponível do *Scrum*.

4.2 MEMBROS DO PROJETO

O destaque das metodologias ágeis se dá pela grande valorização das interações entre as equipes e clientes, bem como a rápida resposta e absorção das mudanças identificadas como sendo necessárias nessas constantes interações (SILVA; LOVATO, 2016). Logo, nessa categoria temática foi investigada como seria a aplicação do *Scrum* a um projeto solo, visto que se perde essa vantagem de ter as interações entre os membros da equipe e os clientes. Entretanto, no Brasil existem muitas microempresas de *software* com um único desenvolvedor, sendo que cerca de 60% das empresas de *software* consolidadas no mercado, iniciaram suas atividades com somente um desenvolvedor (PAGOTTO *et al.*, 2016). Nessa categoria temática foi

levantada características específicas e únicas de um projeto de *software*, que permite a eliminação de uma equipe de projeto. Os entrevistados ressaltaram que o desenvolvedor deve ter experiência para solucionar problemas sozinho, capacidade de lidar com as pessoas, além disso, é imprescindível que o mesmo faça uma coleta e uma análise de requisitos bem-feita no início do projeto, além de um cronograma bem definido.

Todos os entrevistados citaram de formas diferentes que as principais vantagens da equipe de projeto seriam as trocas de conhecimento, as resoluções de problemas de forma mais facilitada, ter mais pontos de vista, o que levaria a alternativas melhores, a diversidade de habilidades e até mesmo uma produtividade maior, desde que a divisão seja organizada. Sobre as vantagens de se ter uma equipe de projetos um dos entrevistados ressaltou:

> "Supondo que é uma boa equipe, a principal vantagem é que um membro pode ajudar o outro em caso de necessidade. Outra vantagem é que dá pra dividir todo o trabalho e reduzir o tempo de desenvolvimento. Além disso, vale ressaltar que é possível (embora inviável em vários casos) ter um especialista para cada área do desenvolvimento (Ex. Um expert em *front-end*, um expert em *back-end*, um expert em banco de dados, etc.)."

Em contrapartida, algumas desvantagens do desenvolvimento de projetos em equipe, que são solucionadas ou amenizadas com o desenvolvimento solo são: dificuldade de gerenciamento de equipes, dependência entre os membros para executar as atividades, conflitos e divergências de opiniões, dificuldades de comunicação e transmissão de conhecimentos. Sobre comunicação foi obtido o seguinte relato:

> "É necessária uma comunicação muito assertiva para não ocorrer inconsistências no processo de desenvolvimento, por exemplo, dois desenvolvedores trabalhando na mesma funcionalidade, mas que entenderam a funcionalidade de forma diferente. Outra desvantagem é que se a equipe for boa, mas tiver um membro ruim responsável por uma funcionalidade importante, ele sozinho pode gerar problemas sérios no projeto."

4.3 *SCRUM MASTER* E *PRODUCT OWNER*

Com relação às funções de *Scrum Master* e de *Product Owner* alguns apontamentos podem ser realizados. Ao adaptar o *Scrum* para o desenvolvimento solo, essas funções não podem ser inseridas da forma que a literatura descreve, não existindo pessoas específicas para desenvolver essas funções, mas sim uma pessoa que acumulará essas funções. O ponto principal discutido com relação a falta desses papéis foi a falta de pluralidade de visões, discutida também na categoria membros do projeto.

> "A colaboração entre papéis diferentes costuma ser ótima para melhorar o processo de software. Ter perspectivas diversas ajuda a encontrar soluções diferentes para os problemas e evita que você fique preso em suas próprias ideias. Mesmo que esteja trabalhando sozinho, é importante buscar feedback externo e se conectar com outros profissionais para obter opiniões diferentes e garantir que suas decisões sejam sólidas."

Para eliminar essas funções, os entrevistados, em sua maioria, afirmaram que é imprescindível que o desenvolvedor tenha capacidades interpessoais, ou seja, seja capaz de se relacionar com os clientes, estabelecendo uma comunicação assertiva com os mesmos.

> "Para ser um Product Owner dedicado, é necessário estabelecer uma comunicação direta e regular com o cliente. Consulte-o em decisões importantes, priorize suas necessidades e obtenha feedback contínuo. Mantenha registros claros e trabalhe de forma iterativa. O objetivo é entender as necessidades do cliente e garantir que o produto final atenda às suas expectativas."

4.4 *BACKLOG DO PRODUTO (PRODUCT BACKLOG)*

Com relação a criação do *Product Backlog*, o estabelecimento das prioridades e da colocação de novos itens a maioria dos entrevistados acreditam que essas atividades são mais fáceis quando o projeto é solo, pois não é necessário lidar com opiniões diversas de um número maior de pessoas, tendo o desenvolvedor autonomia para tomar decisões rapidamente. Acredita-se que no projeto solo é possível focar mais nas necessidades do projeto e dos usuários, entretanto ressaltaram que o acerto no estabelecimento de prioridades envolve experiência no gerenciamento desse tipo de projeto, além disso, o desenvolvedor pode ter uma visão influenciada pelo cliente, que muitas vezes pode achar que alguns requisitos são mais importantes do que outros sem um critério técnico, o que pode gerar requisitos, funcionalidades que não acrescentam valor ao projeto. Um dos entrevistados ressaltou a relação direta e simplificada que o desenvolvedor solo possui ao fazer o *Product Backlog*.

> "A relação com o *Product Backlog* é direta e simplificada. Ele tem autonomia para adicionar novos itens e realizar alterações conforme necessário, sem a necessidade de negociação com a equipe. O desenvolvedor tem total visibilidade do *backlog* e é responsável por sua atualização após cada *Sprint*. Essa abordagem proporciona agilidade e autonomia no processo de desenvolvimento de *software*."

4.5 *SPRINT*

No desenvolvimento solo, as entregas ao final de cada *Sprint* podem ocorrer de diferentes maneiras, como a entrega ao cliente de forma presencial ou remota, além disso, o desenvolvedor pode solicitar ao cliente que teste o que foi entregue como um protótipo e forneça feedbacks para ajustes a serem implementados. A abordagem escolhida depende do contexto e dos acordos com o cliente. A comunicação clara e transparente com o cliente é fundamental para garantir que as entregas atendam às expectativas e estejam alinhadas com as metas do projeto. A forma de entregar pode ser variável, como descrita por um dos entrevistados.

> "As entregas podem ser de forma presencial, remota, o desenvolvedor pode entregar um protótipo e pedir para o cliente preencher um formulário. Enfim, cada desenvolvedor vai definir isso de acordo com a sua experiência, o tipo de cliente e o tipo de produto. Isso gera uma vantagem, que é o fato de o desenvolvedor obter o *feedback* diretamente do cliente, sem riscos de que esse *feedback* seja alterado até chegar nele."

4.6 TEMPO E AGILIDADE

Todos os entrevistados acreditam que a utilização do *Scrum*, mesmo que de forma solo, pode auxiliar na agilidade do projeto. Ressaltaram de formas diferentes que o *Scrum* divide e organiza o processo de desenvolvimento, isso fornece uma diretriz para que o desenvolvedor possa seguir. Ao mesmo tempo, é importante ressaltar que o desenvolvedor precisa ter muita disciplina e responsabilidade para seguir o *Scrum* sozinho, uma vez que ninguém (além do cliente) irá cobrá-lo para executar o projeto. Algumas características do *Scrum* que permitem que o projeto seja executado de forma mais ágil: trabalho mais visível, saber o que fazer primeiro e ter *feedbacks* constantes. Com relação ao cronograma um dos entrevistados relatou:

> "O uso do Scrum facilita a observação de como está o cronograma do projeto, pois se alguma delas sofrer atraso, percebe-se mais rapidamente, uma vez que o desenvolvimento da Sprint é mais rápido. Em metodologias mais tradicionais, muitas vezes só se percebe o atraso em fases mais avançadas do projeto, uma vez que muitas coisas precisam ser desenvolvidas antes de serem entregues."

Outra questão levantada referente a questão de tempo e agilidade são as reuniões diárias e retrospectivas. Acredita-se que no desenvolvimento solo o desenvolvimento pode ser mais ágil ao eliminar essas reuniões, porém muitos destacaram em suas respostas que o desenvolvedor deve adaptar essas etapas separando um tempo para refletir sobre as questões do projeto, como quais tarefas já foram feitas, quais ainda precisam ser feitas, se há atraso ou não, entre outras.

4.7 SATISFAÇÃO DO CLIENTE

A maioria dos entrevistados defendem que a utilização do *Scrum* e também de suas adaptações a projetos de desenvolvimento solo pode aumentar a satisfação do cliente. As suas características que mais contribuem para que isso aconteça são as entregas incrementais e visíveis, pois as *Sprints* são rápidas e permitem que o cliente acompanhe o progresso, a flexibilidade em ajustar prioridades e requisitos com base no retorno do cliente, além da comunicação frequente e transparente. Foi possível verificar isso no seguinte relato:

> "As *Sprints* são rápidas, o cliente terá um produto com alguma funcionalidade utilizável muito rápido, com isso ele pode participar do processo, verificar mais facilmente o andamento do projeto e fornecer um *feedback* ao desenvolvedor sem a necessidade de implementar muita coisa de uma vez (o que tornaria as mudanças mais complexas). Além disso, dependendo do projeto, alguma funcionalidade mais rápida já pode resolver alguns problemas do cliente sem que ele tenha que esperar o desenvolvimento total do produto."

4.8 ASPECTOS GERAIS

Ao longo da análise das informações, alguns aspectos gerais foram levantados. Esses aspectos visam a concluir sobre a avaliação, viabilidade, vantagens, desvantagens e fatores de influência na aplicação do *Scrum* em projetos solo. De acordo com a visão e experiência dos entrevistados o *Scrum* funciona muito bem quando aplicado da forma correta, sendo que as *Sprints* foram consideradas a principal vantagem do *framework.* Em geral elas facilitam o processo de criação de *software* "utilizável" rapidamente, além disso, facilita as alterações e mudanças, pois essas alterações podem ser feitas em estágios do desenvolvimento, que não necessariamente vão afetar um bloco muito grande do sistema. Também facilita o acompanhamento do projeto com relação ao cronograma, dessa forma, a maioria dos entrevistados defenderam que o *Scrum* é bem adaptável a diversos tipos de projeto, inclusive desenvolvimento solo.

> "Com ou sem uma equipe, ele pode ajudar a ter mais organização, priorizar o trabalho e manter o foco no que é importante. O *Scrum* permite uma entrega incremental do *software* e envolve o cliente de forma colaborativa. É uma maneira de ser ágil e adaptável mesmo trabalhando sozinho. No geral, acho que o *Scrum* aplicado a projetos solo de desenvolvimento pode trazer benefícios e facilitar o processo."

Alguns pontos que precisam ser observados com relação a adaptação da metodologia foram: encontrar maneiras de obter *feedback* externo para validar decisões, buscar oportunidades de aprendizado e desenvolvimento profissional e garantir uma boa comunicação e colaboração com o cliente. Além disso, é importante estar aberto a ajustes e melhorias contínuas no processo. Também ressaltaram a importância de se ter a documentação do projeto, mesmo que não seja o foco, sendo uma das sugestões o uso de um quadro *kanban* para acompanhamento das tarefas que precisam ser feitas, das que estão em andamento e as que já foram feitas e testadas, bem como as que já foram implantadas.

5. DISCUSSÃO DOS RESULTADOS

As entrevistas mostraram que o *Scrum* e os princípios ágeis podem ser adaptados e utilizados na gestão solo de *softwares*. Neste tópico, os resultados apresentados foram discutidos em função da literatura, de forma a destacar a avaliação e análise de possíveis

aplicações do *Scrum* e de princípios ágeis na gestão de projetos solo de *softwares.* Ao analisar as respostas da primeira categoria temática, pode-se inferir que o *Scrum* é adaptável. De acordo com as respostas, o *mesmo* possui características intrínsecas que permitem adaptações aos projetos solo de desenvolvimento de *softwares*. Algumas etapas descritas na literatura não podem ser aplicadas, mas não prejudicam o gerenciamento do projeto e podem ser substituídas por outras ações do desenvolvedor.

Com relação aos membros do projeto, é inegável as vantagens de uma equipe, entretanto é possível suprir a falta da mesma quando o desenvolvedor possui experiência para resolver sozinho possíveis problemas que possam surgir, além disso, deve possuir habilidades interpessoais e de comunicação. Destaca-se também a possibilidade de eliminar as funções de *Product Owner* e *Scrum Master*, sendo que esses papéis serão realizados pelo desenvolvedor, que buscará gerenciar o projeto e avaliar as atividades que foram desenvolvidas.

Ao analisar a categoria *Product Backlog*, percebe-se as vantagens do desenvolvimento solo. Decisões podem ser tomadas de forma mais rápida com foco nas reais necessidades do cliente, visto que o desenvolvedor receberá retornos diretamente. Entretanto, ressalta-se novamente a necessidade de um desenvolvedor experiente capaz de ser assertivo na definição do *Product Backlog*, a definição correta das listas de atividades, funcionalidades, características, melhorias, requisitos e regras é primordial para garantir o sucesso do projeto. Nas *Sprints* o desenvolvedor possui autonomia e as entregas podem se dar de formas variadas a depender do tipo de produto e cliente.

Outros pontos importantes a serem destacados é que todos os entrevistados acreditam que o uso de princípios e metodologias ágeis influenciam no tempo e na agilidade do projeto. As entregas incrementais que podem ser realizadas tanto em projetos formados por equipes quanto no desenvolvimento solo são grandes diferenciais quando comparado com as metodologias tradicionais. No desenvolvimento solo destaca-se uma visão simplificada e eficiente do cronograma do projeto, além da eliminação das reuniões diárias e retrospectivas que na prática podem tornar o projeto mais ágil, consequentemente as entregas rápidas, incrementais e visíveis foram fatores citados que aumentam a satisfação do cliente. Nos aspectos gerais é importante ressaltar que a utilização eficaz do *Scrum* em projetos solo depende do desenvolvedor que deverá buscar *feedbacks* externos e oportunidades de aprendizado e desenvolvimento profissional. Por fim, não é possível eliminar o processo de documentação, mas este pode ser simplificado quando aplicado ao gerenciamento e desenvolvimento solo de projetos. Com base nas informações discutidas, o Quadro 2 apresenta um resumo dos resultados encontrados.

Quadro 2 - Resumo dos resultados

Fatores analisados	Resultados
1. Viabilidade de adaptação do *Scrum* a projetos de desenvolvimento solo	O *Scrum* pode ser adaptado e usado de forma eficiente em projetos de desenvolvimento solo.
2. Membros do projeto	É importante que o desenvolvedor tenha experiência em desenvolvimento de *softwares*
	Necessário que o desenvolvedor tenha habilidades interpessoais
	Necessidade de uma comunicação assertiva com o cliente
3 Scrum Master e Product Owner	Possibilidade de eliminar as funções de: Product Owner e *Scrum Master*, sendo que esses papéis serão realizados pelo desenvolvedor.
4. Product backlog	Decisões rápidas e controladas pelo desenvolvedor
	Agilidade no processo de estabelecer prioridades e acréscimo de novos itens
	Foco nas necessidades do projeto e do cliente
	É importante que o desenvolvedor tenha experiência em desenvolvimento solo para ser assertivo no *Product Backlog*.
5. *Sprint*	Entregas baseadas na experiência do desenvolvedor, tipo de cliente e o tipo de produto
	Autonomia do desenvolvedor
	Necessidade de uma comunicação assertiva com o cliente
6. Tempo e agilidade	Entregas mais rápidas e incrementais
	Imprescindível que o desenvolvedor seja organizado, disciplinado e tenha responsabilidade com o projeto
	Visão simplificada e eficiente do cronograma do projeto
	Eliminação das reuniões diárias e retrospectivas, que podem tornar o projeto mais ágil
7. Satisfação do cliente	Aumento de satisfação do cliente com a utilização do *Scrum* e de metodologias ágeis, inclusive no desenvolvimento solo.
	Aumento da satisfação devido as entregas rápidas, incrementais e visíveis
8. Aspectos Gerais	Necessário encontrar maneiras de obter *feedback* externo para validar decisões
	Necessidade de buscar oportunidades de aprendizado e desenvolvimento profissional
	Não é possível eliminar o processo de documentação, mas é possível que este seja simplificado.

Fonte: Elaborado pela autora (2024)

6. CONCLUSÃO

Esta pesquisa teve como objetivo analisar as aplicações do *Scrum* e de princípios ágeis na gestão de projetos solo de *softwares*. Para isso, primeiramente foi realizada uma pesquisa sobre metodologias ágeis, posteriormente foi feito um levantamento de pesquisas existentes que aplicaram princípios e metodologias ágeis em projetos de desenvolvimento solo de *softwares*.

Foi realizado um estudo de caso descritivo que consistiu na realização de entrevistas semiestruturadas com seis profissionais. A partir das entrevistas foi possível criar um quadro com os resultados, que consistiu em levantar aspectos específicos da aplicação do *Scrum* em projetos solo a partir de categorias temáticas pré definidas: percepção sobre o *Scrum*, membros do projeto, *Scrum Master* e *Product Owner*, *Product Backlog*, *Sprint*, tempo e agilidade, satisfação do cliente e aspectos gerais. Em linhas gerais, a partir das experiências e visões dos entrevistados foi possível concluir que é viável a aplicação do *Scrum* e de princípios ágeis na gestão de projetos solo de *softwares*, além disso, os objetivos específicos descritos na introdução foram alcançados. Entretanto, pontos relevantes como a experiência do desenvolvedor e suas habilidades interpessoais e de comunicação são fatores indispensáveis.

Sugere-se um novo estudo comparando a aplicação do *Scrum* em projetos de desenvolvimento de *software* solo e projetos desenvolvidos por equipes.

REFERÊNCIAS

ABRAHAMSSON, P.; WARSTA, J.; SIPONEN, M.T.; RONKAINEN, J. New directions on agile methods: a comparative analysis. *In*: **25TH INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 2003.** Proceedings. Portland, OR, USA: IEEE, p. 244–254, 2003. Disponível em: <http://ieeexplore.ieee.org/document/1201204/>. Acesso em: 7 mar. 2024.

BASTOS, Almiro Midons; BASTOS, Argemiro Midonês. Uso do Scrum como método para otimização na elaboração de projetos. **Research, Society and Development**, v. 10, n. 9, p. 1-11, 2021.

BECK, K.; BEEDLE, M.; BENNEKUM; A.; COCKBURN, A.; CUNNINGHAM, W.; FOWLER, M.; GRENNING, J.; HIGHSMITH, J.; HUNT, A.; JEFFRIES, R; KERN, J.; MARICK, B.; MARTIN, R.C.; MELLOR, S.; SCHWABER, K.; SUTHERLAND, J.; THOMAS, D. **Manifesto for Agile Software Development**. 2001. Disponível em: <http://agilemanifesto.org/>.

DENNING, S. Why Agile can be a game changer for managing continuous innovation in many industries. **Strategy & Leadership**, v. 41, n. 2, p. 5–11, 2013.

FADEL, A. C.; SILVEIRA, H. M. **Metodologias ágeis no contexto de desenvolvimento de software:** XP, Scrum e Lean. 2010. Trabalho de conclusão de curso - Universidade Estadual de Campinas. Limeira, p. 26. 2010.

GIL, A. C. **Como elaborar projetos de pesquisa**. 4. ed. São Paulo: Atlas, 2008.

HIGHSMITH, J. ***Agile Project Management: Creating Innovatine Products***. Addison Weley Longman Publishing, 2004.

LAKATOS, E. M; MARCONI, M. **Fundamentos da Metodologia Científica**. 5. ed. São Paulo: Atlas, 2003.

LAVILLE, C.; DIONNE, J. **A construção do saber**: manual de metodologia da pesquisa em ciências humanas. Belo Horizonte: Editora UFMG, 1999.

MARTINS, L.; ROCHA, M.; SANTOS, C.; SAVOINE, M. **Análise de Gerenciamento de Projeto de Software utilizando Metodologia Ágil XP e Scrum**: Um estudo de caso Prático. *In:* **ENCOINFO - CONGRESSO DE COMPUTAÇÃO E TECNOLOGIAS DA INFORMAÇÃO**, 11., 2009, Palmas - TO. Anais. Palmas - TO: CEULP/ULBRA, 2009. p. 149 - 158.

MENDOZA, D. M. Q.; PIZZOLATO, N. D. Gerenciamento e planejamento de projetos de software usando metodologias ágeis: um estudo de caso**. Revista De Gestão e Secretariado**, v.14, n.11, p.1-18, 2023.

MINAYO, M. C. S. O desafio do conhecimento: pesquisa qualitativa em saúde. 9. ed. São Paulo: Hucitec, 2006.

PAGOTTO, T.; FABRI, J.A; LERARIO, A.; GONÇALVES, J.A. **Scrum solo**: Software process for individual development. In: **11TH IBERIAN CONFERENCE ON INFORMATION SYSTEMS AND TECHNOLOGIES (CISTI)**. Espanha: IEEE, 2016. p. 1–6.

PMI. **Um Guia do Conhecimento em Gerenciamento de Projetos (Guia PMBOK®)**. 5. ed. *Project Management Institute*, 2013.

POUDEL, R., GARCIA DE SOTO, B., MARTINEZ, E. Last Planner System and Scrum: Comparative analysis and suggestions for adjustments. **Front. Eng. Manag**., v.7, 359–372, 2020.

PRESSMAN, R. S. **Engenharia de *Software***. 6. ed. São Paulo: McGraw-Hill, 2006.

PREVIATO, E. V. **Uma abordagem de desenvolvimento solo de aplicações utilizando princípios ágeis**. 2018. Dissertação (Mestrado em Ciência da Computação) – Universidade Federal de São Carlos, São Carlos, 2018. Disponível em: https://repositorio.ufscar.br/handle/ufscar/11137

PRODANOV, C. C.; FREITAS, E. C. **Metodologia do trabalho científico:** métodos e técnicas da pesquisa e do trabalho acadêmico. 2.ed. Rio Grande do Sul: Universidade Feevale, 2013.

RISING, L.; JANOFF, N. S. The Scrum Software Development Process for Small Teams. **IEEE Software**, v.4, 26-32, 2000.

SAMPIERI, R. H. **Metodologia de pesquisa**. 3.ed. São Paulo: McGraw-Hill, 2006.

SBROCCO, J. H. T. C.; MACEDO, P. C. **Metodologias ágeis**: engenharia de software sob medida. 1. ed. São Paulo: Érica, 2012.

SCHWABER, K.; SUTHERLAND, J. **O Guia Definitivo para o *Scrum***: As Regras do Jogo. 2016. Disponível em: <http://www.scrumguides.org/docs/scrumguide/v2016/2016-Scrum-Guide-US.pdf>. Acesso em: 29 de mar. de 2024.

SILVA, A. C. **Sistema de Gerenciamento de Tarefas para usuários de *Scrum***. 2011. 115 p. Dissertação (Graduação em Engenharia Eletrônica e Computação) – Escola Politécnica – Departamento de Eletrônica e Computação, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2011. Disponível em: < http://www.repositorio.poli.ufrj.br/monografias/monopoli10002482.pdf >

SILVA, E. C. Da; LOVATO, L. A. Framework *Scrum*: Eficiência em Projetos de Software. **Revista de Gestão e Projetos**, v. 07, n. 02, p. 01–15, 2016.

SILVA, R. E. da; SOUZA NETO, J. Contratação do desenvolvimento ágil de *software* na administração pública federal: riscos e ações mitigadoras. **Revista do Serviço Público Brasília**, v.66, n. 1, 97-120, 2015.

SOMMERVILLE, I. **Engenharia de software**. 9. ed. São Paulo: Pearson, 2011.

SUTHERLAND, J. ***Scrum****: a arte de fazer o dobro de trabalho na metade do tempo*. 1. ed. Editora Leya: 2014.

XU, Y.; KOIVUMÄKI, T. Digital business model effectuation: An agile approach. **Computers in Human Behavior**, v. 95, p. 307–314, 2019.