INTERFACE TECNOLÓGICA

O SCRUM NA PERSPECTIVA DO DESENVOLVEDOR DE SOFTWARE

SCRUM FROM A SOFTWARE DEVELOPER'S PERSPECTIVE

Ingrid Stein Nassr – msteinnassr@outlook.com
Faculdade de Tecnologia de Taquaritinga (Fatec) – Taquaritinga – SP – Brasil

Daniela Gibertoni – daniela.gibertoni@fatec.edu.br
Faculdade de Tecnologia de Taquaritinga (Fatec) – Taquaritinga – SP – Brasil

DOI: 10.31510/infa.v19i2.1494
Data de submissão: 01/09/2022
Data do aceite: 28/11/2022
Data da publicação: 20/12/2022

RESUMO

Com o constante crescimento do mercado, ocorreu uma evolução no gerenciamento de desenvolvimento de software tendo em vista que métodos antigos não eram mais adequados às necessidades do cliente e do desenvolvimento do produto, fazendo surgir novos métodos ágeis. Diante disso, este trabalho apresenta a relevância do Scrum como ferramenta de gestão de projetos a partir de um estudo de caso feito em uma empresa de desenvolvimento de um software denominada Future Tech. Para a fundamentação teórica, inicia-se com uma revisão bibliográfica do tema, baseada em livros e documentações. No estudo de caso, serão apresentadas características de cada processo de desenvolvimento mostrando as suas vantagens no gerenciamento de um projeto como, por exemplo, melhoria em qualidade, priorização das tarefas importantes, tempo de entrega, divisão de tarefas e diminuição nas tarefas que precisam ser refeitas.

Palavras-chave: Scrum. Metodologia Ágil. Desenvolvedor. Gestão de projetos.

ABSTRACT

With the constant growth of the market, there was an evolution in the management of software development, considering that old methods were no longer adequate to the needs of the customer and the development of the product, giving rise to new agile methods. Therefore, this work presents the relevance of Scrum as a project management tool based on a case study carried out in a software development company called Future Tech. For the theoretical foundation, it begins with a bibliographic review of the theme, based on books and documentation. In the case study, characteristics of each development process will be presented, showing its advantages in managing a project, such as improvements in quality, prioritization of important tasks, delivery time, division of tasks and reduction in tasks that need to be redone.

Keywords: Scrum. Agile Methodology. Developer. Project Management.

1 INTRODUÇÃO

De acordo com Sbrocco e Macedo (2012), as metodologias ágeis surgiram pela necessidade de buscar alternativas para os modelos tradicionais de desenvolvimento de projetos. Um dos pontos que mais era discutido é que as metodologias tradicionais em projetos pequenos e de curta duração poderiam atrasar. As metodologias ágeis assumem que os projetos sejam entregues e funcionais o quanto antes na medida em que partes do seu desenvolvimento forem concluídas.

O Scrum foi desenvolvido no início dos anos 90 por Jeff Sutherland e Ken Schwaber, não apenas para desenvolvimento de software, mas também para vários setores da economia pelo simples fato de ser um método inovador e simples, potencializando um dos maiores bens da modernidade: o tempo.

O presente artigo tem como objetivo discutir um projeto prático com o uso do Scrum como ferramenta de gestão de projetos em uma empresa denominada Future Tech, utilizando para tal dos procedimentos metodológicos apresentados a seguir.

Desta forma, está estruturado em cinco seções, sendo que a primeira delas a introdução, a qual apresenta a contextualização da proposta pesquisada. Na seção dois é descrita a fundamentação teórica contendo as informações sobre o tema central da pesquisa. Na seção três é apresentada a metodologia que foi utilizada para realização da pesquisa. Na seção quatro é apresentado o estudo de caso, retratando a visão de um desenvolvedor de software que utilizou a metodologia o Scrum em um projeto. E, por fim, na seção cinco são apresentados os resultados obtidos com a pesquisa.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção são apresentados o conceito referente ao tema da pesquisa deste artigo e o Scrum, obtidos através da pesquisa bibliográfica em livros e *sites*.

2.1 METODOLOGIAS ÁGEIS

Segundo Pricklandnicki, Willi e Milani (2014), em meados dos anos 90, surgiram processos alternativos de desenvolvimento de software em resposta aos tradicionais,

considerados excessivamente regrados, lentos, burocráticos e inadequados à natureza da atividade.

O termo metodologia ágil ficou conhecido em 2001 quando dezessete especialistas em processos de desenvolvimento de software que já utilizavam “métodos leves”, reuniram-se e criaram modelos comuns a partir de princípios básicos que eram respeitados por todos quando tinham sucesso a um projeto.

O resultado foi a criação da Aliança Ágil e do "Manifesto Ágil", os quais incentivam o uso de melhores práticas de desenvolvimento de software, contendo vários princípios que definem critérios para o processo de desenvolvimento ágil.

De acordo com Gomes (2013), “Scrum, Extreme Programming (XP), Crystal Clear e Feature Driven Development são exemplos de métodos ágeis. Cada um deles traz uma abordagem diferente que inclui diversos valores, práticas e reuniões”.

Porém, o processo ágil apresenta quatro valores:

- Indivíduos e interações, ao contrário de processos e ferramentas.
- Software executável, ao contrário de documentação.
- Colaboração do cliente, ao contrário de negociação de contratos.
- Respostas rápidas a mudanças, ao contrário de seguir planos.

Além dos quatro valores, segundo Sbrocco e Macedo (2012), o processo é formado por 12 princípios:

- Entrega adiantada do software de qualidade;
- Os requisitos mudam de acordo com o agrado do cliente;
- Entrega de software frequentemente funcionando;
- Pessoas de negócios e desenvolvedores devem trabalhar juntos;
- Time sempre motivado;
- Conversar com a equipe é essencial;
- Software sempre funcionando;
- Ritmo constante entre todos os envolvidos no projeto;
- Excelência técnica e bom design aumentam a agilidade;
- Maximizar a quantidade de trabalho não realizado;
- Os melhores projetos surgem de equipes auto-organizáveis;
- Em intervalos regulares a equipe se reúne para se tornar mais eficaz.

Segundo Gomes (2014), uma pesquisa feita pela VersionOne.com mostra os seguintes benefícios após a implantação de metodologias ágeis em projetos de desenvolvimento de software:

- Melhor *Time-to-market* e Maior Retorno sobre o Investimento;
- Maior Satisfação do Cliente e Melhor Gestão de Mudanças de Prioridades;
- Melhor Visibilidade dos Projetos;
- Maior produtividade;
- Equipes mais motivadas;
- Melhor Disciplina na Engenharia e Melhor Qualidade Interna;
- Processo de Desenvolvimento Simplificado;
- Redução de Risco;
- Redução de Custos.

Segundo Sbrocco e Macedo, “um aspecto muito importante que diferencia a metodologia ágil das tradicionais é que os métodos ágeis utilizados são orientados a pessoas e não a processos”, isso indica que essa metodologia foi criada para fugir dos processos metódicos ditados por sequencias exaustivas de tarefas que levam a um produto, substituindo-a por uma metodologia crítica que observa e age de acordo com o necessário, tornando assim o desenvolvimento do projeto mais produtivo, autônomo, simples e satisfatório para a equipe e ao cliente.

2.2 *SCRUM*

No início dos anos 90, Jeff Sutherland enfrentou diversos problemas para fazer as coisas com mais rapidez e eficiência. As equipes trabalhavam de forma contraditória entre si, acabava sendo um processo lento não entregando o que tinha sido proposto e tendo como consequência a perda de tempo e dinheiro. Jeff Sutherland junto com Ken Schwaber então criaram um método inovador, o Scrum, que vem simplificando a vida, o trabalho e o tempo não somente no desenvolvimento de software, mas também nos setores da saúde, finanças, ensino superior e telecomunicações (SUTHERLAND, 2016).

Conforme o Scrum Guide (2020), pode-se dizer que o Scrum é um “framework leve que ajuda pessoas, times e organizações a gerar valor por meio de soluções adaptativas para problemas complexos”.

Segundo Gomes (2014), “o Scrum, traz uma abordagem voltada para a gestão, com maior foco nas reuniões, no planejamento e na melhoria contínua.”.

É uma metodologia baseada no empirismo, que afirma que o conhecimento vem da experiência e da tomada de decisões baseadas nas observações e no *lean thinking* que reduz o desperdício e concentra-se no essencial.

Segundo Sommerville (2011), o Scrum possui três fases:

> A primeira é uma fase de planejamento geral, em que se estabelecem os objetivos gerais do projeto e da arquitetura do software. Em seguida, ocorre uma série de ciclos de *sprint*, sendo que cada ciclo desenvolve um incremento do sistema. Finalmente, a última fase do projeto encerra o projeto, completa a documentação exigida, como quadros de ajuda do sistema e manuais do usuário, e avalia as lições aprendidas com o projeto.

Os profissionais trabalham de forma conjunta para resolver problemas trazendo resultados para a empresa. Trabalha com uma equipe reduzida onde cada membro tem uma função e responsabilidade para entregar mais em menos tempo. Tem como objetivo com que os projetos sejam mais ágeis e eficientes para assim obter sucesso nas suas ações. Também ajuda a organizar processos, identificando e eliminando possíveis problemas, sendo assim pode-se identificar mais claro os possíveis problemas do projeto.

2.2.1 *SCRUM TEAM*

O *Scrum Team* é um pequeno time de pessoas que possui um *Product Owner*, um *Scrum Master* e os *Developers*. Dentro do time não há hierarquias, constitui-se apenas em profissionais focados em um objetivo de cada vez para garantir a meta do produto.

Jeff Sutherland e Ken Schwaber (2020), descobriram que times menores atuam em melhor comunicação e produtividade sendo assim são formados normalmente por 10 ou menos pessoas.

2.2.2 *PRODUCT OWNER*

Segundo Schwaber e Sutherland (2020), um projeto ágil só é bem-sucedido se tiver um bom *Product Owner*. Ele representa o profissional que tem a visão do que será desenvolvido, as necessidades que devem ser atendidas, o público que irá utilizar o produto e

os objetivos a serem alcançados. É o responsável em sobreavaliar o trabalho do time e gerenciar o *backlog* do produto que inclui os seguintes pontos:

- Desenvolver e comunicar explicitamente a meta do produto;
- Criar e comunicar claramente os itens do *Product Backlog*;
- Ordenar os itens do *Product Backlog*; e,
- Garantir que o *Product Backlog* seja transparente, visível e compreensível.

Um excelente *Product Owner* deve ter as seguintes qualidades:

- Saber como gerenciar as expectativas dos stakeholders e suas prioridades;
- Ter uma visão clara e um bom conhecimento do produto;
- Saber coletar requisitos, para que assim possa transformar a visão de produto em um bom backlog;
- Estar disponível para engajar ativamente com a equipe;
- Ser um bom organizador que consiga fazer múltiplas atividades;
- Saber comunicar a visão do produto;
- Ser um bom líder, treinando, guiando e dando suporte para toda a equipe.

Segundo Schwaber e Sutherland (2020):

> Para que os Product Owners tenham sucesso, toda a organização deve respeitar suas decisões. Essas decisões são visíveis no conteúdo e na ordem do Product Backlog e por meio do incremento inspecionável na revisão da sprint.

2.2.3 *SCRUM MASTER*

Scrum Master é a figura central responsável para garantir que todos entendam e sigam a cultura ágil conforme definido no Guia do Scrum tornando o time cada vez mais independente, auxilia o *Product Owner* com sugestões e melhorias no gerenciamento do *backlog*.

Segundo Sabbagh (2013), "o *Scrum Master* está presente e age como um facilitador em todas as reuniões do *Scrum*, facilita o trabalho do dia a dia do time de desenvolvimento e facilita as interações entre o time de desenvolvimento e o *Product Owner*". Além disso, possui habilidades suficientes para desenvolver, testar e criar, ou seja, tudo que for necessário para entregar o produto funcionando corretamente.

2.2.4 *DEVELOPERS*

Segundo Sabbagh (2013), o "time de desenvolvimento é um grupo multidisciplinar de pessoas, responsável por realizar o trabalho de desenvolvimento do produto.".

Executa o trabalho que foi planejado, cria a *sprint backlog* e é responsável por todas as atividades relacionadas ao produto. São responsáveis por:

- Criar um plano para a *Sprint*, o *Sprint Backlog*;
- Introduzir gradualmente qualidade aderindo a uma definição de pronto;
- Adaptar seu plano a cada dia em direção à meta da *Sprint*;
- Responsabilizar-se mutuamente como profissionais.

Portanto cabe aos desenvolvedores as funções de monitorar a qualidade e o desenvolvimento técnicos do produto, que deverá receber *feedbacks* sobre melhorias a cada *sprint*, assim como planejar as tarefas que deverão ser feitas para que se atinja o objetivo visado para o produto.

2.3 EVENTOS

Segundo Schwaber e Sutherland (2020), a *Sprint* é um contêiner para todos os outros eventos. Cada evento no *Scrum* é uma oportunidade formal para inspecionar e adaptar os artefatos do *Scrum*. Esses eventos são projetados especificamente para permitir a transparência necessária.

2.3.1 *SPRINT*

A *Sprint* é um ciclo de desenvolvimento onde acontece a *Sprint Planning*, *Daily Meeting*, *Sprint Review* e *Sprint Retrospective*. Cada *Sprint* tem a duração de duas a quatro semanas e elas acontecem até a finalização do projeto, são eventos de duração fixa com objetivo de gerar incremento. Ela é composta pelos seguintes eventos: *planning, daily meetings, review* e *retrospective* (SBROCCO; MACEDO, 2016).

Segundo Schwaber e Sutherland (2020), não se faz mudanças que coloquem em risco a meta da *sprint*, possui foco na qualidade, refinamento de requisitos conforme o necessário e somente o *Product Owner* pode cancelar a *sprint*.

2.3.2 *SPRINT PLANNING*

Segundo Sabbagh (2013), a *Sprint Planning* é o evento que inicia a *sprint* e serve para definir o *Sprint Backlog*, ou seja, uma lista de tarefas que serão realizadas durante um período que julgam ser capazes de desenvolver naquela *sprint*.

A *sprint planning* possui uma duração máxima de oito horas para *Sprint* de um mês e para *Sprint* mais curtas, o evento tem uma duração menor (SCRUM GUIDE, 2020).

Esse evento responde as seguintes questões fundamentais:

- Por que esta *Sprint* é valiosa?
- O que pode ser entregue como resultado ou incremento?
- Como será realizado?

Pelo exposto acima, a *sprint planning* tem a função de organizar e temporizar as tarefas que serão desenvolvidas e acompanhadas na *sprint,* esse método aumenta a produtividade, pois melhora o planejamento de toda a equipe.

2.3.3 *DAILY SCRUM*

De acordo com o Scrum Guide (2020), *Daily Scrum*, também conhecida como *Daily Meeting*, são reuniões diárias com duração aproximadamente de 15 a 20 minutos para compartilhar o progresso, planejar o próximo dia e identificar impedimentos. Cada membro responde as seguintes perguntas:

- O que fiz ontem?
- O que farei hoje?
- Existe algo me impedindo?

Segundo Sabbagh (2013), a *Daily Scrum "*produz um plano informal para o próximo dia de trabalho do time de desenvolvimento, ou seja, para até a próxima reunião de *Daily Scrum."*.

Conforme o Schwaber e Sutherland (2020), "as *Daily Scrums* melhoram as comunicações, identificam os impedimentos, promovem a rápida tomada de decisões e consequentemente, eliminam a necessidade de outras reuniões.".

Portanto a *daily* é de suma importância para organização e o desenvolvimento do projeto, pois com ela é possível manter o foco nas prioridades e na qualidade do produto que está sendo desenvolvido, deixando o time como uma unidade coesa.

2.3.4 *SPRINT REVIEW*

É o momento de entregar o trabalho que foi desenvolvido ao dono do produto tendo a possibilidade de ele aceitar ou não o que foi feito. Também é quando se determina adaptações e ajuste do *backlog*, se necessário. Caso o dono do produto não aceitar será necessário realizar um novo ciclo para entregar um produto na qual ele aceite e só assim será possível passarmos para o próximo e último evento da *Sprint* (SCHWABER; SUTHERLAND, 2020).

Segundo Schwaber e Sutherland (2020), a *Sprint Review* é o penúltimo evento da *Sprint* e tem duração de no máximo quatro horas.

2.3.5 *SPRINT RETROSPECTIVE*

Conforme o Scrum Guide (2020), a *Sprint Retrospective* é o último evento da *Sprint* e serve para refletir como melhorar o processo de trabalho da equipe para a próxima *Sprint* focando sempre na qualidade e eficácia.

É voltada para inspecionar a si próprio e criar um plano de melhoria para a próxima *Sprint*. O evento costuma ter a duração de no máximo três horas. Somente o time de *Developers* entra na discussão de como foi o trabalho do time já o *Scrum Master* tem o papel de fazer o evento ocorrer e garantir que todos os participantes entendam o seu propósito sempre de maneira positiva e produtiva (SCHWABER; SUTHERLAND, 2020).

- Identificar e ordenar os principais itens que foram bem e as potenciais melhorias que devem ser implementadas;
- Criar um plano para implementar melhorias no modo que o time de Scrum faz o seu trabalho;

Ao final da *Retrospective* o time *Scrum* deverá ter identificado melhorias que serão implementadas na próxima *sprint* (SABBAGH, 2013).

3 PROCEDIMENTOS METODOLÓGICOS

O presente trabalho foi realizado em duas etapas baseadas em procedimentos metodológicos diferentes.

A primeira etapa é composta de uma revisão bibliográfica baseada em artigos e livros que abordam o tema central da pesquisa, o Scrum.

A segunda etapa é composta de um estudo de caso baseado em uma experiência vivenciada na empresa denominada Future Tech que utiliza o Scrum como metodologia ágil em um de seus times para o desenvolvimento de software.

4 RESULTADOS E DISCUSSÃO

O estudo de caso foi realizado com um desenvolvedor *backend* em um projeto de desenvolvimento de uma aplicação mobile e uma web de uma empresa denominada Future Tech.

O processo começou com um levantamento de dados, analisando as práticas do *Scrum* no projeto e suas circunstâncias, dando ênfase na aplicação dessa metodologia ágil no dia a dia. Isso possibilitou reconhecer algumas falhas, como por exemplo: a inclusão de várias tarefas no meio da *Sprint* e o revezamento de funcionalidade de cargos, isto é, o líder técnico do time elege alguém para direcionar a reunião, ficando vago a função de *Scrum Master*, já que o comando torna-se rotativo a cada *sprint*.

Atualmente, o time é composto pela equipe de *backend* contendo seis pessoas, portal (*frontend*) com oito pessoas, mobile com sete pessoas, QA com três pessoas, UX/UI com duas pessoas e o líder técnico/DevOps.

As *Daily Meetings* são realizadas três vezes na semana com equipes separadas e duas vezes na semana feitas com o time todo, sempre no mesmo horário, busca-se otimizar o período de duração da reunião a partir da divisão dos times, em que quando há o encontro coletivo, cada membro da equipe mostra a visão geral que seu time percorreu no processo.

A *Sprint Planning* é o primeiro evento da *Sprint* e funciona da seguinte maneira: cada equipe faz o levantamento das tarefas levando em consideração que nenhuma tarefa de outra equipe fique bloqueada. Normalmente, ela é conduzida pelo membro de cada equipe e é feita com *planning poker*, que é uma ferramenta de estratégia utilizada para estimar a duração de cada tarefa, após isso elas são lançadas por esse mesmo membro no JIRA, ferramenta de monitoramento de tarefas e acompanhamento de projetos, para que o resto do time acompanhe o desenvolvimento do projeto.

A *Sprint Review* acontece no final da *Sprint* apresentando o que foi desenvolvido e normalmente quem conduz é a equipe do portal (*frontend*) e mobile.

A *Sprint Retrospective* é a última etapa da *Sprint* sendo conduzida pelo líder técnico ou por algum outro membro do time.

A metodologia *Scrum*, utilizada no projeto funcionou muito bem obtendo muitos resultados positivos, como por exemplo: a rápida execução e entrega das tarefas tendo como resultado o adiantamento de futuras tarefas, as complicações ou *bugs* foram corrigidos assim que foram identificados facilitando as *Sprints* seguintes, a otimização do tempo de trabalho deixa o time mais coeso e comunicativo, isso motiva os desenvolvedores porque é possível notar com clareza as progressões feitas em seu trabalho.

Outro aspecto relevante é o método de testagem a cada tarefa executada, a partir de uma equipe criada exclusivamente para testes, os desenvolvedores mandam suas tarefas executadas para testagem e recebem um retorno do time responsável por isso, sendo assim é muito mais rápido, simples e eficaz a correção de erros por etapa, deixando pouca abertura para erros difíceis de serem encontrados e criando assim um código funcional. O projeto também recebe um teste final e, com frequência, não há erros no momento da execução total porque todos foram previstos e solucionados na testagem por etapas.

Mas um aspecto negativo de grande peso durante o estudo de caso foi o acumulo de duas funções, *Product Owner* e *Scrum Master*, em um única pessoa porque o mesmo não se posicionava efetivamente, durante o apontamento de problemas, o líder colocava as decisões a seres tomadas para a equipe que ficava sem um direcionamento efetivo, tendo que criar e gerenciar suas próprias soluções, essa situação gerava alguns conflitos acerca das soluções que deveriam ser tomadas e impedia a visão de alguns problemas futuros. A centralidade em uma figura com perfil de liderança ajudaria a gerenciar o tempo e a estratégia nos momentos de resoluções de problemas.

5 CONSIDERAÇÕES FINAIS

O presente trabalho buscou esclarecer a importância do *Scrum* como ferramenta de gestão de projetos em uma empresa na área de desenvolvimento de software, iniciando com a fundamentação teórica baseada na revisão bibliográfica seguindo de um estudo de caso baseado em uma experiência vivenciada por um desenvolvedor *backend* na empresa Future Tech.

Mediante o conhecimento adquirido por meio das pesquisas bibliográficas, foi possível observar neste estudo de caso que a metodologia *Scrum* pode apresentar falhas, principalmente pelo fator humano caso alguma etapa do processo for mal executada ou definidas incorretamente. Apesar de ser uma forma interessante de fazer a gestão de projetos, pode não apresentar resultados esperados se não forem devidamente orientados a exercerem seus devidos papéis. Não existe uma metodologia ágil melhor ou pior para se utilizar, e sim aquela que atende melhor o projeto conforme as suas necessidades. Depois de fazer um levantamento de requisitos é possível identificar qual metodologia atende melhor, principalmente considerando os conhecimentos obtidos através desse artigo.

Por fim, pode-se dizer que pelo olhar de um desenvolvedor que o uso do *Scrum* em um projeto foi positivo para o restante do time e para o desenvolvimento da aplicação porque foi possível que o time visse seu trabalho progredindo e mantendo-se sempre motivados, já para a aplicação, quase nunca apresentava erro em produção, pois antes eram testados e validados para garantir um código funcional. A metodologia pode apresentar falhas, principalmente quando não há uma divisão clara das funções do líder e o mesmo acumula duas delas, bem como a falta de posicionamento do mesmo na gestão de problemas e possíveis soluções. Sendo assim, desde que as funções dos trabalhadores fiquem bem definidas e executadas, os resultados serão sempre conforme o esperado ou até melhores, já que a metodologia Scrum apresentou no estudo de caso mais aspectos positivos do que negativos, sendo os negativos falhas em relação à gestão de pessoas e não ao método propriamente dito.

REFERÊNCIAS

GOMES, ANDRÉ FARIAS. **Agile Desenvolvimento de Software com Entregas Frequentes e Foco no Valor de Negócio.** São Paulo: Casa do Código. 2014.

PRIKLANDNICKI, R; WILLI, R; MILANI, F. **Métodos Ágeis Para Desenvolvimento de Software.** Porto Alegre: Bookman. 2014.

SABBAGH, RAFAEL. **Scrum. Gestão Ágil para Projetos de Sucesso.** São Paulo: Casa do Código. 2013.

SBROCCO, José Henrique Teixeira de Carvalho; MACEDO, Paulo Cesar. **Metodologias ágeis: Engenharia de Software Sob Medida.** São Paulo, Érica. 2012.

SCHWABER, K.; SUTHERLAND, J. **Scrum Guide.** 2020. Disponível em: https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-PortugueseBR-3.0.pdf

SOMMERVILLE, Lan. **Engenharia de Software.** 9. ed. São Paulo: Pearson, 2011.

SUTHERLAND, JEFF. **Scrum a Arte de Fazer o Dobro.** São Paulo: Leya. 2016.