Utilização do *Scrum* no desenvolvimento de uma aplicação *web*: um Estudo de Caso

Thales F. Dal Molim[1], **Francisco Carlos M. Souza**[1]

[1]Universidade Tecnológica Federal do Paraná (UTFPR)
Dois Vizinhos – PR – Brasil

Abstract. *Scrum stands out for its ability to deal with the unpredictability inherent in complex projects, therefore, one of the concepts used is that of self-managed and multifunctional teams. However, when ensuring greater autonomy to a team, greater commitment is required, especially in uncertain environments, such as the academic environment. Thus, this article aims to demonstrate the application of Scrum in the development of a web application in an academic environment, through a case study. After the execution, it was concluded that, although the essential requirements were delivered, initially there were moments of high productivity, however, during the execution there were delays and unproductiveness, mainly due to the lack of regular meetings, which led to a demotivation of the team.*

Resumo. *O Scrum destaca-se pela capacidade de lidar com a imprevisibilidade inerente a projetos complexos, para tanto, um dos conceitos utilizados é o de equipes autogeridas e multifuncionais. Entretanto, ao garantir maior autonomia a uma equipe, requer-se um maior comprometimento da mesma, especialmente em ambientes incertos, como o meio acadêmico. Desta forma, o presente artigo tem por objetivo demonstrar a aplicação do Scrum no desenvolvimento de uma aplicação web em ambiente acadêmico, por meio de um estudo de caso. Após a execução, concluiu-se que, embora os requisitos essenciais tenham sido entregues, inicialmente houve momentos de alta produtividade, porém, ao longo da execução houve atrasos e improdutividade, devido principalmente à falta de reuniões regulares, que levaram a uma desmotivação da equipe.*

1. Introdução

Desenvolver softwares é uma tarefa complexa. De um modo geral, esta complexidade está relacionada à própria natureza dos sistemas de software, isto é, não são regidos pelas leis da física nem por processos fabris, não havendo limites naturais para seu potencial. Dada sua inerente complexidade, em pouco tempo tornam-se difíceis de entender e de modificar [Schwaber 2004, Sommerville 2011].

De acordo com [Sommerville 2011], a Engenharia de Software surge para auxiliar no desenvolvimento profissional de software de qualidade, oferecendo, para tanto, diferentes processos que devem ser adequados ao contexto específico de um determinado projeto. Sob o mesmo ponto de vista, para [Pressman and Maxim 2015], o processo não deve ser rígido, mas adaptável ao problema, ao projeto, à equipe e à cultura organizacional aos quais será inserido.

Neste contexto, as metodologias ágeis destacam-se por serem capazes de lidar com projetos complexos, uma vez que são capazes de controlar a imprevisibilidade íntrinseca

a estes projetos. Para tanto, devem se adaptar ao projeto em que estão inseridas, garantindo entregas frequentes por meio de uma série de incrementos, onde cada incremento corresponde a uma nova funcionalidade do sistema.

Sendo assim, o presente trabalho pretende demonstrar a utilização de uma metodologia ágil por meio de um estudo de caso em ambiente acadêmico, visando descrever as atividades e relatar as vantagens ou mesmo desvantagens encontradas no processo. Para tanto, utilizou-se o *framework Scrum* para desenvolver uma aplicação *web* com o objetivo de facilitar trocas e empréstimos de livros entre usuários dentro de uma rede de consumo colaborativo.

O restante do artigo está estruturado da seguinte maneira: a seção 2 descreve o *framework Scrum* e seu funcionamento; a seção 3 apresenta o estudo de caso e as atividades nele desenvolvidas; e, por fim, a seção 4 apresenta as considerações finais.

2. Scrum

De acordo com [Rubin 2012], a origem do *Scrum* pode ser traçada desde um artigo de 1986 [Takeuchi and Nonaka 1986] que evidenciava a importância de equipes multidisciplinares e autogeridas, utilizando o termo *"scrum"* em referência às equipes do *rugby*, propondo que as equipes permaneçam intactas do começo ao fim e sejam responsáveis por combinar as diversas etapas de desenvolvimento, criando, portanto, uma certa continuidade.

Baseando-se nesta premissa, Jeff Sutherland e sua equipe criam, nos anos 1990, o *framework Scrum* para desenvolvimento de software. Em 1995, Ken Schwaber publica o primeiro artigo apresentando o processo então criado [Schwaber 1995]. Nos próximos anos, Sutherland e Schwaber realizariam diversas publicações referentes ao *Scrum* [Rubin 2012].

O *Scrum*, na definição de [Schwaber and Sutherland 2017], é um *framework* estrutural capaz de lidar com problemas complexos, visando entregar maior valor em menor tempo. Para tanto, o *Scrum* define papéis, eventos e artefatos que devem ser associados ao time e são essenciais ao processo.

2.1. Papéis, eventos e artefatos

O desenvolvimento inicia-se com a definição do *Product Backlog*, artefato composto por uma lista de requisitos funcionais e não funcionais que devem materializar os anseios dos *stakeholders*. O responsável por gerenciar este artefato é o *Product Owner (PO)*, que deve, também, garantir que o Time de Desenvolvimento compreenda com clareza os itens listados, visando otimizar o valor do trabalho desenvolvido.

O Time de Desenvolvimento é multifuncional e autogerido, ou seja, a equipe possui todas as competências necessárias para realizar o trabalho, bem como a liberdade para decidir a melhor forma de realizá-lo, sem interferências externas. Um papel chave no andamento do projeto, é o *Scrum Master (SM)*. É o responsável pela manutenção correta do processo, implementando e ensinando as práticas do *Scrum* a todos os envolvidos no projeto. O *SM* deve garantir que as reuniões ocorram corretamente e que a equipe entenda seu propósito.

A natureza do *Scrum* é iterativa, sendo assim, o trabalho flui por meio de uma sequência de iterações, chamadas *Sprints*. Cada *Sprint* se inicia com uma reunião de Planejamento da *Sprint*, onde planeja-se quais funcionalidades do *Product Backlog* serão implementadas, as quais constituirão um novo artefato: o *Sprint Backlog*. São então realizadas reuniões diárias (*Daily Scrum*) para verificar o andamento da *Sprint*. Ao final da *Sprint* é realizada a *Sprint Review*, uma reunião mais longa que visa conferir o que foi realizado até então, os problemas ocorridos, bem como discutir o que poderá ser feito a seguir [Schwaber and Sutherland 2017].

Finalmente, outro artefato utilizado é o *Burndown Chart*, uma representação gráfica que visa representar a quantidade de trabalho restante a ser desenvolvido em uma *Sprint*, no eixo vertical, e o tempo restante, no eixo horizontal. Por meio deste artefato, é possível mensurar a produtividade da equipe, alertar possíveis atrasos, e calcular quando o trabalho será finalizado [Schwaber 2004].

3. Estudo de Caso

A pesquisa coordenada e organizada por [Failla 2016] na 4ª edição de Retratos da Leitura no Brasil, aponta dados importantes sobre hábitos de leitura no país. Da amostra utilizada na pesquisa, 56% dos entrevistados considera-se leitor(a)[1], crescimento de 6% em comparação à mesma pesquisa realizada em 2011, denotando um aumento no número de brasileiros que se identificam como leitores.

Dentre os que se consideram leitores, a pesquisa [Failla 2016] elencou também as razões para não se ter lido mais, das quais evidenciam-se os seguintes números: 8% não leu mais devido à falta de bibliotecas perto de onde mora; 7% porque acha o preço dos livros caro; 5% porque não tem dinheiro para comprar; e 3% por não ter um local próximo de onde mora para comprar livros, como mostra a Figura 1.

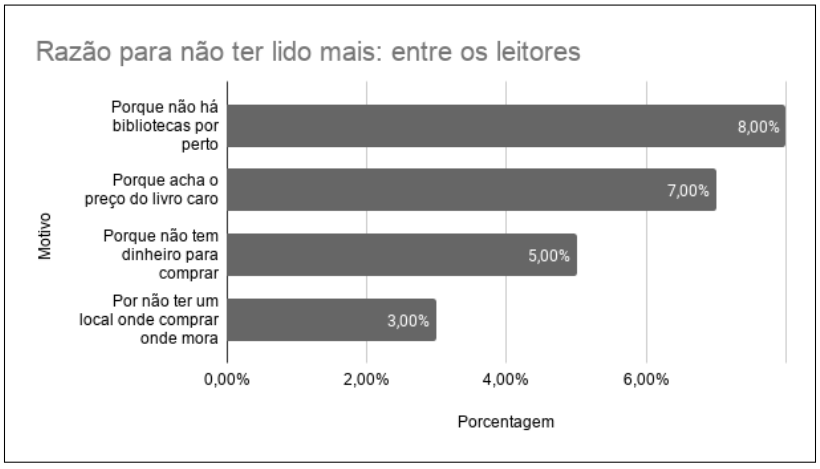

Figura 1. Razão para não ter lido mais: entre os leitores [Failla 2016]

Portanto, temos que 23% dos leitores identificados leram menos devido a motivos relacionados à condição financeira ou pela falta de proximidade de bibliotecas ou

[1]De acordo com o critério utilizado por [Failla 2016], define-se leitor como sendo aquele que leu, inteiro ou em partes, pelo menos 1 livro nos 3 meses anteriores à pesquisa.

lojas físicas. Outro dado relevante constatado pela pesquisa [Failla 2016] é o fato de que 81% dos leitores são usuários da *internet*. Sendo assim, entendemos que a *internet* pode constituir um meio efetivo de atingir o leitor e influenciar no hábito da leitura.

Desta forma, levando em consideração os dados levantados em relação às razões que levaram os leitores a lerem menos, e em relação ao consumo de *internet* pelos mesmos, foi proposto o desenvolvimento de uma ferramenta *online* que visa facilitar a aquisição de livros, via trocas e empréstimos entre os próprios usuários, a ser desenvolvida utilizando o *framework Scrum*.

3.1. Proposta

A ferramenta proposta foi nomeada *Escambook* (aglutinação das palavras "escambo" e *"book"*) e baseia-se no princípio do consumo colaborativo, definido por [Botsman and Rogers 2011] como sendo um modelo de consumo baseado na colaboração entre membros de uma comunidade, podendo ser algo local ou através da *internet*, que busca formar grupos de pessoas interessadas em compartilhar bens, por meio de trocas, empréstimos ou mesmo doações.

Considerando os dados levantados anteriormente em relação ao uso da *internet* por leitores no Brasil, em conjunto com dados levantados por [Wroblewski 2011] referentes ao crescente uso de *smartphones* pela população global, a ferramenta foi proposta dentro do paradigma *web*, utilizando o conceito *mobile first*[2].

3.2. Planejamento

Por tratar-se de um projeto desenvolvido como parte de um trabalho a ser entregue visando a conclusão de uma disciplina acadêmica, o *Scrum* precisou ser adaptado para tanto. O desenvolvimento deveria durar, no máximo, o período de um semestre letivo, correspondendo ao tempo da disciplina ministrada. Portanto, as *Sprints* foram previamente planejadas da seguinte maneira: três *Sprints* iniciais com a duração de uma semana cada, e duas *Sprints* finais com a duração de duas semanas cada.

A equipe, composta por seis alunos, todos com conhecimento prévio acerca do *framework* utilizado, foi dividida devidamente em: um *PO* e cinco desenvolvedores, sendo um destes o *SM*. Foram criadas contas, por cada membro da equipe, nas plataformas: *Slack*, para auxiliar na comunicação; *Trello*, para simular um quadro de tarefas e visualizar o fluxo de trabalho; e *GitHub*, para controlar o versionamento do produto. Além destas, outras ferramentas auxiliares foram utilizadas pela equipe, como exposto na Tabela 1.

[2]Abordagem na qual o desenvolvimento de aplicações tem foco direcionado a dispositivos móveis [Wroblewski 2011].

Tabela 1. Principais ferramentas utilizadas pela equipe

Ferramenta	Objetivo
draw.io	Elaborar diagramas, dentre outras representações de alto nível.
GitHub	Controlar o versionamento do produto.
Google Drive	Armazenar, criar e editar documentos e artefatos.
Insomnia	Testar as requisições feitas à API.
pgAdmin 4	Gerenciar o banco de dados do aplicativo.
Slack	Auxiliar na comunicação da equipe.
Trello	Simular um quadro de tarefas para visualizar o fluxo de trabalho.
Visual Studio Code	Produzir o código-fonte do aplicativo.

Devido ao ambiente em que o projeto estava inserido, as reuniões do *Scrum* precisaram ser adaptadas, de modo que: o Planejamento das *Sprints* deveria ocorrer na primeira aula da semana correspondente à *Sprint*, juntamente com a *Sprint Review* da *Sprint* anterior. Já as reuniões diárias, ocorreram informalmente e somente quando possível, devido a restrições relacionadas ao horário disponível de cada membro da equipe

Após definido o escopo do projeto, o *PO* em conjunto com a equipe levantou os requisitos funcionais e não funcionais que comporiam o *Product Backlog*. Foi criada então uma pasta compartilhada em um serviço de armazenamento (*Google Drive*), onde cada membro teria acesso a este e aos demais artefatos.

Para a implementação, optou-se pela composição de duas aplicações distintas: uma aplicação *frontend*, responsável pela interação do usuário com o produto, e uma API *backend*, responsável por gerenciar a interação entre o *frontend* e a base de dados. Foi escolhido o padrão MVC (*model-view-controller*[3]) para estruturar o aplicativo como um todo, como demonstrado na Figura 2.

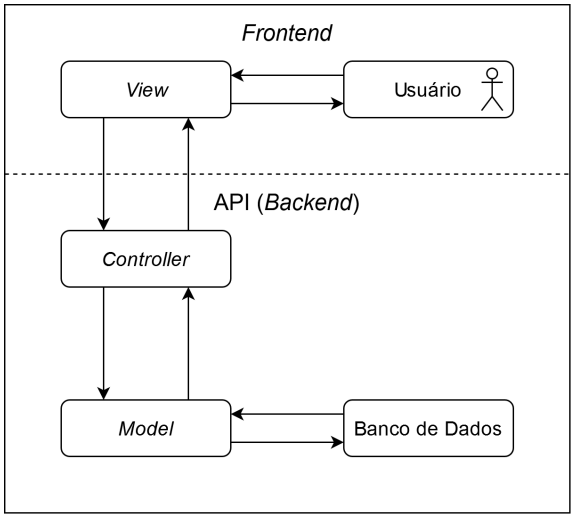

Figura 2. Arquitetura do aplicativo proposto

[3]Padrão de arquitetura de software na qual a interface de usuário é desacoplada das funcionalidades da aplicação e dos dados armazenados [Pressman and Maxim 2015].

Para o desenvolvimento da aplicação *frontend*, decidiu-se pela utilização do *Vue.js*, *framework JavaScript* de código-aberto, englobando, portanto, além da linguagem *JavaScript*, a linguagem de marcação HTML e a linguagem de folhas de estilo CSS. Para a implementação da API *backend*, utilizou-se *Node.js*, interpretador de *JavaScript* de código-aberto, associado ao SGBD *PostgreSQL*, também de código-aberto.

3.3. Primeira *Sprint*

A primeira *Sprint* foi utilizada para o planejamento geral do projeto, nela foram levantados os requisitos iniciais e determinados os aspectos técnicos, como as tecnologias e padrões a serem utilizados. Foram iniciados, também, os trabalhos de documentação, elaboração de diagramas, bem como foram levantados e reunidos materiais de estudo objetivando nivelar a equipe quanto ao conhecimento técnico.

Decidiu-se, também, por dividir o Time de Desenvolvimento em dois: uma equipe para desenvolver a aplicação *frontend* e outra para desenvolver a API *backend*, pois alguns integrantes não eram familiarizados com algumas das tecnologias empregadas, assim cada equipe poderia ter um foco de estudo mais direcionado. Para tanto, a comunicação entre as duas equipes deveria ser constante. Não houveram grandes dificuldades, todos empenharam-se e entregaram o que havia sido estipulado no Planejamento da *Sprint*.

3.4. Segunda *Sprint*

Para a segunda *Sprint*, foram alocadas seis funcionalidades a partir do *Product Backlog*, nesta iteração surgiram as primeiras dificuldades. Embora a documentação e os papéis estivessem bem definidos, o desenvolvimento falhou em grande parte devido à inexperiência de alguns integrantes acerca das tecnologias empregadas, como fora apontado pelos mesmos na *Sprint Review*. Das seis funcionalidades que deveriam ser entregues, a equipe do *backend* concluíra quatro, ficando o *frontend* inteiramente por fazer (Tabela 2).

Tabela 2. Funcionalidades a serem entregues na segunda *Sprint*

ID	Nome	*Frontend*	*Backend*
RF001	Cadastrar usuário	Pendente	Entregue
RF005	Fazer login	Pendente	Entregue
RF006	Fazer logout	Pendente	Entregue
RF011	Cadastrar livro	Pendente	Entregue
RF015	Visualizar mural de livros	Pendente	Pendente
RF016	Acessar biblioteca pessoal	Pendente	Pendente

3.5. Terceira *Sprint*

No planejamento da terceira *Sprint*, definiram-se três novas funcionalidades a serem desenvolvidas, além das que permaneceram incompletas na *Sprint* anterior. Desta vez, a comunicação e o desenvolvimento em equipe fluiram melhor, havendo maior envolvimento do time nas tarefas. O *backend* concluiu o que lhe fora designado, enquanto o *frontend* deixou apenas uma funcionalidade por fazer, como visto na Tabela 3.

Sendo assim, embora o desempenho da *Sprint* tenha sido no geral positivo, os integrantes responsáveis pelo *frontend* destacaram, na *Sprint Review*, que não haviam se

dedicado o quanto gostariam, pois, embora as funcionalidades tivessem sido entregues, não estavam satisfeitos com a qualidade de seu trabalho.

Tabela 3. Funcionalidades a serem entregues na terceira *Sprint*

ID	Nome	*Frontend*	*Backend*
RF001	Cadastrar usuário	Entregue	-
RF005	Fazer login	Entregue	-
RF006	Fazer logout	Entregue	-
RF011	Cadastrar livro	Entregue	-
RF015	Visualizar mural de livros	Entregue	Entregue
RF016	Acessar biblioteca pessoal	Entregue	Entregue
RF017	Adicionar livro na biblioteca pessoal	Entregue	Entregue
RF018	Remover livro da biblioteca pessoal	Entregue	Entregue
RF019	Marcar ou desmarcar livros disponíveis	Pendente	Entregue

3.6. Quarta *Sprint*

Na quarta *Sprint* foram alocadas as últimas quatro funcionalidades previstas, além da que havia ficado para trás na última *Sprint*. Esta seria a primeira *Sprint* dentre as duas de maior duração (duas semanas). O número reduzido de funcionalidades para um maior período de tempo se justifica pelo fato de estas serem funções mais complexas do aplicativo, como o registro de trocas e empréstimos de livros entre usuários.

Porém, esgotadas as duas semanas, o *backend* havia implementado todas as funcionalidades, enquanto o *frontend* havia entregue apenas uma (Tabela 4). Na *Sprint Review*, concluiu-se que, além de tudo, o *frontend* do aplicativo estava deixando a desejar em qualidade, e propôs-se que, estando o *backend* concluído, na próxima *Sprint*, que seria utilizada para melhorias e correções gerais, as duas equipes se reuniriam para uma refatoração total do *frontend*, caso contrário o produto final não atingiria as expectativas.

Ainda durante a *Sprint Review*, os próprios integrantes da equipe responsável pelo *frontend* admitiram que faltou comprometimento e dedicação de sua parte, apesar dos constantes estímulos e cobranças do *SM*. Dessa forma, percebeu-se também que, separar o Time de Desenvolvimento em duas equipes pode não ter sido uma boa experiência, pois enquanto uma das equipes mostrou total dedicação ao projeto, a outra deixou a desejar.

Tabela 4. Funcionalidades a serem entregues na quarta *Sprint*

ID	Nome	*Frontend*	*Backend*
RF014	Buscar livros	Entregue	Entregue
RF019	Marcar ou desmarcar livros disponíveis	Pendente	-
RF024	Registrar troca	Pendente	Entregue
RF027	Registrar empréstimo	Pendente	Entregue

3.7. Quinta *Sprint*

A quinta e última *Sprint*, foi, portanto, voltada para uma refatoração do *frontend*. Constatou-se que a parte mais complexa acabou sendo implementada pelo *SM*, que

possuía maior conhecimento técnico, delegando as tarefas menos complexas ao restante do time, bem como, reaproveitando o possível do que já havia sido produzido, uma vez que faltava pouco tempo para a entrega final do produto.

Ao final das duas semanas, tanto o *backend* quanto o *frontend* estavam finalizados. Porém, faltou tempo para possíveis melhorias que agregariam maior valor ao produto. Nesta última *Sprint Review*, o time concordou que a comunicação entre as equipes poderia ter sido melhor, e que a falta de reuniões diárias (*Daily Scrum*), que ocorriam apenas quando possível, prejudicou o projeto, pois o desenvolvimento por vezes não manteve constância.

3.8. *Product Release*

A entrega final do produto consistiu em uma apresentação do processo de desenvolvimento do mesmo, seguida de uma demonstração de seu uso, realizadas em sala de aula pelo *PO* e pelo *SM* aos demais integrantes da disciplina.

O aplicativo desenvolvido permite ao usuário, após cadastrar-se no sistema, acessar sua página pessoal, onde encontra-se um dos elementos centrais da plataforma, a biblioteca pessoal (Figura 3), onde poderá visualizar e controlar todos os livros que possui, assim como livros que recebeu através de empréstimos ou trocas com outros usuários.

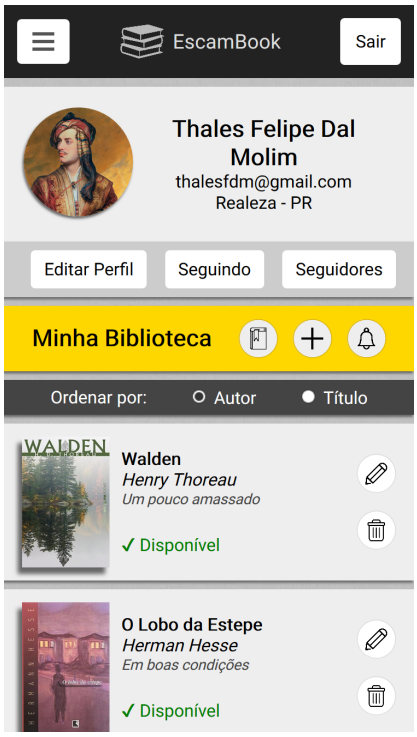

Figura 3. Interface da biblioteca pessoal

A ferramenta consta também com um "mural de livros" (Figura 4), onde é possível buscar livros por título, autor ou ISBN. Através deste, usuários cadastrados podem adicionar à sua biblioteca pessoal cópias que já possuam ou procurar cópias de outros usuários para trocar ou emprestar.

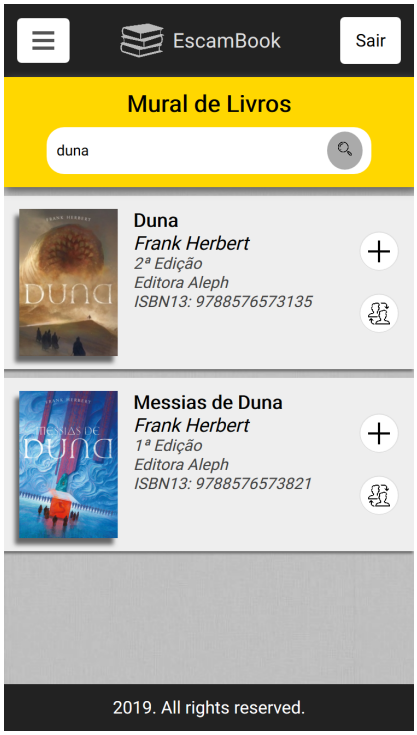

Figura 4. Interface do mural de livros

O produto final atingiu os objetivos propostos, entregando os requisitos essenciais inicialmente propostos no *Product Backlog*. Entretanto, ainda há espaço para aperfeiçoamentos e novas funcionalidades que agregariam maior valor ao produto, como, por exemplo: um *chat*; um sistema de seguidores; um sistema de avaliações; e a busca por usuários de acordo com determinada região geográfica.

4. Conclusão

Buscando demonstrar o uso do *Scrum* no desenvolvimento de uma aplicação *web*, propôs-se a criação de uma ferramenta *online* com o objetivo de facilitar empréstimos e trocas de livros entre usuários, a ser desenvolvida em ambiente acadêmico no período de um semestre letivo.

Inicialmente, foi descrita a origem, a definição e as características do *Scrum*, assim como os papéis, eventos e artefatos que o compõem. Em seguida, foi apresentado o estudo de caso, onde justificou-se a proposta, para que, então, fosse relatado o processo de planejamento e desenvolvimento do aplicativo por meio do *framework Scrum*.

Verificou-se, durante o desenvolvimento, a importância de seguir corretamente as instruções do *Scrum*, uma vez que, o uso dos artefatos e a definição dos papéis de acordo com o perfil de cada integrante manteve a equipe melhor organizada, visto que o *PO* concentrou-se em levantar e gerenciar as funcionalidades do *Product Backlog*, enquanto o *SM* focou-se em manter o funcionamento do processo até o final.

Evidenciou-se, também, que a falta ou irregularidade das reuniões diárias prejudicou o projeto como um todo, visto que houveram picos de produtividade em certos momentos quando a equipe se reunia, em contraste aos momentos de hiato entre os encontros, quando houve procrastinação e improdutividade.

Concluiu-se que, apesar das dificuldades encontradas, o produto final foi entregue dentro do prazo e os requisitos essenciais implementados, e que, portanto, a aplicação do *Scrum* foi bem sucedida. Finalmente, em relação ao produto que foi desenvolvido, considerou-se que o mesmo pode ainda ser estendido com novas funcionalidades que agregariam maior valor ao mesmo.

Referências

Botsman, R. and Rogers, R. (2011). *What's mine is yours: the rise of collaborative consumption*. HarperCollins.

Failla, Z. (2016). *Retratos da leitura no Brasil 4*. Sextante.

Pressman, R. and Maxim, B. (2015). *Software Engineering: a practitioner's approach*. McGraw-Hill Higher Education.

Rubin, K. (2012). *Essential Scrum: a practical guide to the most popular agile process*. Addison-Wesley.

Schwaber, K. (1995). Scrum development process. *Advanced Development Methods*.

Schwaber, K. (2004). *Agile project management with Scrum*. Microsoft Press.

Schwaber, K. and Sutherland, J. (2017). *Guia do Scrum*. (n.p.).

Sommerville, I. (2011). *Engenharia de Software*. Pearson Brasil.

Takeuchi, H. and Nonaka, I. (1986). The new new product development game. *Harvard Business Review*.

Wroblewski, L. (2011). *Mobile First*. A Book Apart.