ANÁLISE COMPARATIVA ENTRE MÉTODOS ÁGEIS E TRADICIONAIS NO DESENVOLVIMENTO DE *SOFTWARE*

Jonatas Bernardes de Oliveira[1]; Lauenia Princia Ferreira da Costa[2] ; Lucas Henrique de Castro Oliveira [3], Rhaellen Lorena de Jesus Gonçalves [4], Maria Aparecida Reis França [5]

[1, 2, 3, 4, 5] Universidade de Uberaba - UNIUBE
jonatas-bernardes@hotmail.com; maria.franca@uniube.br

Resumo

Quando surgiu a demanda por *software* surgiu também a dificuldade em produzi-los com qualidade. Logo criou-se regras e valores a serem seguidos para projetar e desenvolver *softwares* que melhor atendessem às necessidades dos usuários e clientes.

Com essa necessidade de criação de regras e valores surgiram algumas metodologias de desenvolvimento: cascata, prototipação e espiral. Atualmente são nomeadas de metodologias tradicionais, pelo motivo de anos mais tarde terem surgido outras.

Essas outras, chamadas de metodologias ágeis, são compostas por características um pouco diferentes, mas carregam o mesmo objetivo: criar *softwares* de qualidade.

Com o objetivo de estudar a diferença entre os dois tipos de metodologia, tradicional e ágil, desenvolvemos um aplicativo de *chat* utilizando a metodologia cascata e, mais tarde, o mesmo aplicativo utilizando a metodologia Scrum.

Apesar de suas diferenças, ambas as metodologias foram eficientes, mas a Scrum proporcionou um melhor resultado.

Analisamos o desenvolvimento do projeto e a resposta encontrada para que uma metodologia tenha gerado um produto melhor que a outra está na comunicação entre a equipe e o cliente realizada mais frequentemente na Scrum.

Palavras-chave: projeto, planejamento, qualidade do *software.*

1 Introdução

Nas últimas décadas do século XX, período em que as empresas começaram a investir em *softwares*, os profissionais da área de tecnologia não estavam preparados para a alta demanda de desenvolvimento de aplicações, o que resultava, muita das vezes, em um desenvolvimento precário e sem princípios para serem seguidos. Este fato veio a alimentar a chamada "crise do *software*", que teve seu início por volta dos anos de 1970 ocasionada por incidentes de *software*.

Na primavera do ano de 2000, dezessete profissionais que já utilizavam meios diferentes de métodos de planejamento de desenvolvimento de *software* se reuniram, todos com a intenção de debater as principais diferenças entre seus pensamentos, além da forma com que cada um lidava com desenvolvimento de aplicações computacionais.

O resultado da reunião foi o desenvolvimento de um documento chamado de "Manifesto Ágil". Este manifesto é uma declaração de diversas opiniões desses profissionais, que chegaram ao consenso que existem regras e valores primordiais que devem ser seguidos e priorizados para se desenvolver *softwares* com eficiência temporal com qualidade.

O planejamento com usos de métodos distintos para se realizar um objetivo não é

algo novo, pois o ser humano pode pensar de diferentes formas e mesmo assim alcançar o cumprimento do objetivo proposto. É notável e aceitável que muitas vezes podemos ter opiniões diferentes sobre como realizar da melhor maneira uma determinada tarefa a nós proposta. Sobre este fato podemos perceber que não existe em nenhuma parte do mundo alguém que possa afirmar concretamente que um método seja infalível, pois segundo Aristóteles (GRÉCIA, 384 a.C. – 322 a.C.) “Não há um só método para estudar as coisas”.

O uso de métodos ágeis em diversos desenvolvimentos (principalmente de *software*) é amplamente incentivado atualmente em contraposição ao uso de métodos tradicionais. Este artigo tem como objetivo discorrer sobre o tema apresentado se baseando em uma experiência no desenvolvimento de um aplicativo móvel utilizando os dois métodos mencionados.

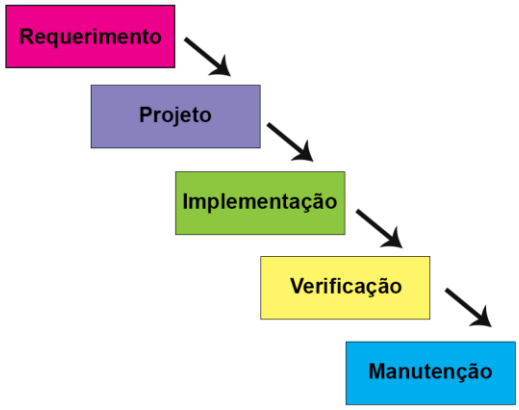
2 Materiais e Métodos

Tanto nas metodologias tradicionais quanto nas metodologias ágeis de desenvolvimento é preciso planejar e só depois executar. Uma das diferenças entre elas é o quanto se planeja antes de executar o desenvolvimento de um *software*. Nas metodologias tradicionais a maior parte do tempo do projeto é reservado para planejar enquanto, nas metodologias ágeis, o planejamento é feito de forma iterativa e incremental, de acordo com as necessidades descobertas pelo caminho (CÁCERES, 2015).

Para comparar o método tradicional com o método ágil de desenvolvimento de *software* selecionamos a mais utilizada entre as metodologias tradicionais e a mais utilizada entre as metodologias ágeis: Cascata e Scrum, respectivamente.

O modelo em cascata é o mais antigo de todos os métodos. Foi criado em 1970 por Winston Walker Royce. O nome se dá devido às suas fases, por serem sequenciais e cascateadas (DIAS, 2019).

Figura 1 - O modelo em cascata.



Fonte: Dos autores baseado em Casa da consultoria (2017).

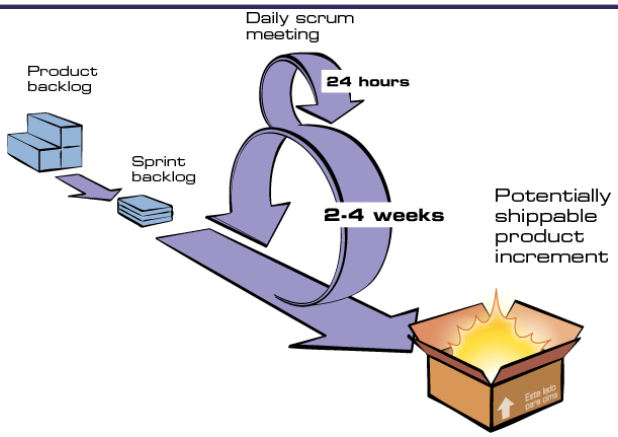
O modelo é composto pelas fases de requerimento, projeto, implementação, verificação e manutenção.

O Scrum é uma metodologia ágil. Foi criado por Jeff Sutherland e Ken Schwaber voltado, inicialmente, para planejar somente projetos de *software*. Atualmente, é utilizado também em outros tipos de projeto, como em otimização no processo de criação e construção de produtos (FONTES, 2019).

Diferente do cascata, o Scrum não é constituído por fases. Os projetos são divididos por ciclos, chamados de *sprints*, que duram entre duas e quatro semanas.

Durante as *sprints* ocorrem planejamentos, reuniões, designação de papéis para os membros da equipe, desenvolvimento de *software* e testes.

Figura 2 - Ciclo Scrum.



Fonte: Desenvolvimento ágil (sem data)

Para analisar o funcionamento das metodologias foram desenvolvidos dois aplicativos de *chat* para a troca instantânea de mensagens. Primeiramente foi desenvolvido seguindo as regras do cascata e, mais tarde, foi desenvolvido o mesmo aplicativo seguindo as regras do scrum.

Desenvolvimento em Cascata

Requerimento

Nessa parte foi feita uma análise e definição de requisitos. Um questionário foi cuidadosamente elaborado, pelo motivo de ser a única vez que teríamos contato com o cliente durante o projeto, e foi feita uma entrevista para descobrir os serviços que precisavam ser fornecidos pelo aplicativo de chat.

Conclui-se que, foi feita uma descrição detalhada do aplicativo e desenhado o diagrama de caso de uso, o diagrama de sequência e o diagrama de classes.

Projeto

Foi iniciada esta etapa pensando na arquitetura do software. Para definir a lógica de programação e os recursos de software que seriam utilizados no projeto, foi desenhado o diagrama de componentes e o diagrama de *container.* Depois disso, foram mapeados os *stakeholders* (todas as pessoas que poderiam estar envolvidas no projeto) e, por fim, foi feito o design das telas.

Implementação

O programador o fez tentando alcançar, ao máximo, os objetivos descritos na fase de requerimento, para atender as expectativas do cliente.

Testes

Com o desenvolvimento finalizado, os desenvolvedores iniciaram os testes para verificar as lógicas internas e funcionalidades externas do aplicativo.

Manutenção

Alguns erros foram descobertos durante os testes e logo foram corrigidos. O aplicativo estava pronto para ser entregue ao cliente.

Desenvolvimento em Scrum

Nada poderia ser iniciado sem antes fazer a análise de requisitos mas, para ser ágil, foi feito um pequeno questionário e uma rápida entrevista com o cliente. Rápido porque, dessa vez, não seria a única vez que ele seria contatado durante o projeto.

Com as respostas em mãos, foi feita uma breve descrição para o aplicativo.

Como sugestão do criadores dos métodos ágeis, foram mapeados os *stakeholders,* foram levantados os riscos (imprevistos que poderiam vir a acontecer) e foram descritas as possíveis soluções para contornar tais riscos.

Em seguida, foram definidos os recursos materiais e recursos humanos que seriam necessários para o projeto.

O Time *Scrum* é composto pelo *Product Owner*, o Time de Desenvolvimento e o *Scrum Master (*SCHWABER; SUTHERLAND*, 2017).*

Dessa forma, foram definidos Jonatas Bernardes como *Scrum master* (o responsável por facilitar reuniões e remover os impedimentos da equipe), Lauenia Ferreira como *Product Owner* (responsável pela qualidade do produto/*software*), Lucas Castro e Rhaellen Lorena como desenvolvedores (responsáveis pelo desenvolvimento do aplicativo).

Para desenvolver uma maior empatia com o usuário, antes de dar início ao desenvolvimento o product owner escreveu algumas *User Stories* (possíveis tarefas que o usuário realizaria no aplicativo) e, em seguida, desenvolveu o *Product Backlog,* que se trata de uma lista de tarefas a serem desenvolvidas durante o projeto.

Conforme mencionado, os projetos em Scrum são divididos em ciclos, e assim foi feito. Foi estipulado que um mês seria o prazo suficiente para finalizar o desenvolvimento do aplicativo, então esse tempo foi dividido em dois ciclos de duas semanas, ou duas *sprints*.

Primeira *Sprint*

Para dar início a *sprint*, o Scrum Master agendou a *Sprint Planning* (reunião de planejamento da *sprint*). Durante a reunião foram selecionados os itens do *product backlog* (lista de tarefas) que achamos que seria possível concluir até o final da sprint e, com o consentimento de toda a equipe, foram criadas as seguintes definições de pronto: o código do aplicativo deveria ser comentado de forma construtiva, para que outros programadores pudessem editá-lo facilmente; quando terminadas as tarefas da lista, o código deveria ser testado pelo programador e depois por outro colega da equipe, para evitar os erros.

Durante a *sprint* foram realizadas as *Daily Meetings* (reuniões diárias), que eram momentos quando todos da equipe contavam ao *Scrum Master* o que havia feito no dia e o que pretendiam fazer no próximo dia, compartilhando os impedimentos, caso houvesse.

No final da *sprint,* todas as tarefas haviam sido concluídas com sucesso e, começou-se os testes.

Os erros descobertos foram corrigidos e então foi apresentado o que havia sido feito para o cliente para saber se ele estava satisfeito e se poderia ser dada continuidade no projeto. Ele ficou satisfeito com o resultado, mas exigiu algumas alterações.

Por fim, foram feitas as reuniões de Sprint Review (reunião para revisar o que foi feito durante a *sprint*) e a *Sprint Retrospective* (reunião para analisar os erros e buscar formas de melhorar).

Segunda *Sprint*

Foi iniciada a segunda *sprint* com a elaboração de um novo, ainda curto, questionário e agendamento de uma nova entrevista com o cliente, para falar sobre as alterações e foram acrescentadas no Product Backlog as novas tarefas.

Durante a Sprint Planning foram selecionadas as tarefas que restaram no product backlog e transferidas para a Sprint Backlog.

Foram feitas novamente as reuniões diárias até concluir a *sprint.* Quando o time de desenvolvimento concluiu o aplicativo, foram feitos novamente os testes e corrigidos os erros.

O resultado foi apresentado ao cliente, que ficou totalmente satisfeito, e foi finalizada a sprint com a Sprint Review, para revisar o que foi feito durante a segunda *sprint* e a *Sprint Retrospective,* para analisar novamente os erros e sugerir melhorias para o próximo projeto.

Cascata ou Scrum?

Apesar de suas diferenças, ambas as metodologias foram eficientes para o desenvolvimento do aplicativo. Portanto, para analisar os resultados, foi preciso definir o que levar em consideração antes de apontar a melhor metodologia.

Foi levado em consideração os seguintes pontos:

- comunicação entre os membros da equipe
- preocupação com a qualidade do software
- preocupação com a usabilidade do *software*
- a qualidade da documentação do *software*

- envolvimento do usuário (cliente) durante o desenvolvimento
- o modo como os *softwares* são testados
- a facilidade ao dar manutenção no software
- o nível de aceitação para alterações de requisitos no software

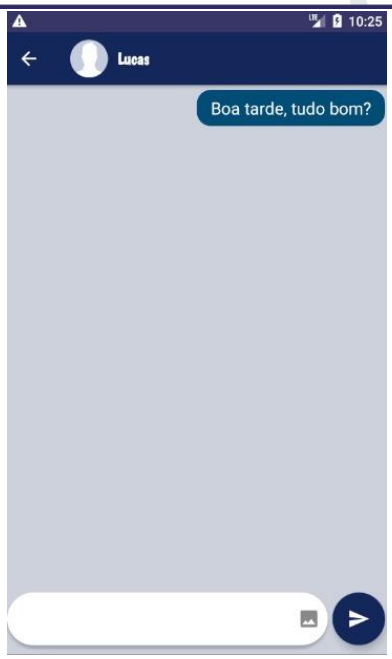
3 Resultados esperados

O desenvolvimento em Cascata foi bastante efetivo e proporcionou ótimos resultados.

Já o desenvolvimento em Scrum foi um pouco mais complexo, mas foi isso que gerou diferença entre os dois aplicativos.

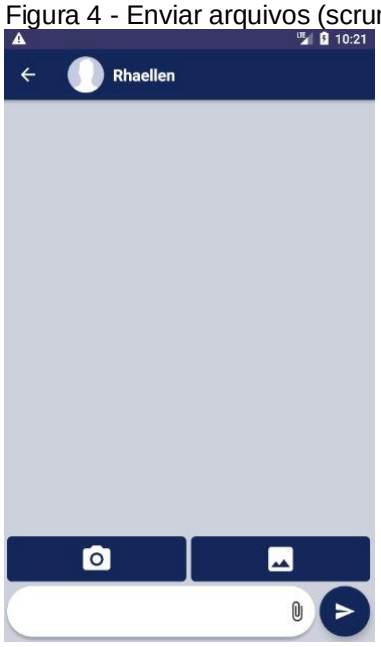
Ambos os aplicativos foram construídos com as opções de realizar cadastro, fazer login, iniciar conversas, visualizar conversas anteriormente iniciadas e anexar imagens da galeria.

Utilizar Scrum como metodologia permitiu acrescentar as funcionalidades de fazer login somente com o webmail da Uniube, de anexar fotos instantâneas da câmera, criar grupos de contatos e uma barra de pesquisa para facilitar a busca por contatos.



Fonte: Dos autores

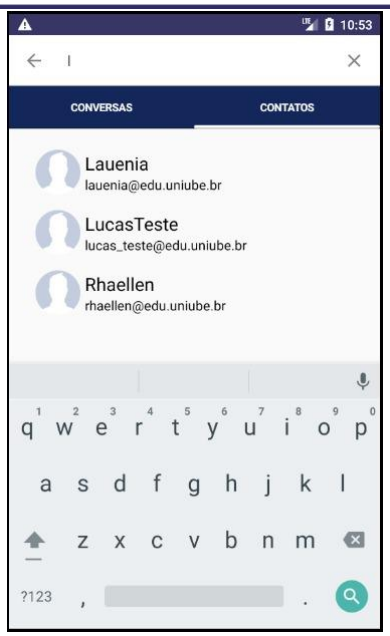
Figura 4 - Enviar arquivos (scrum).



Fonte: Dos autores.

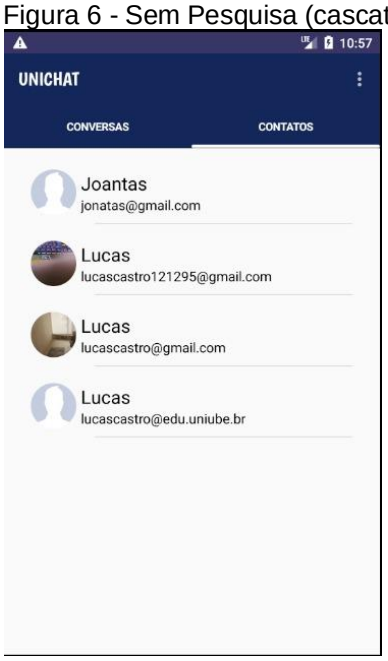
Figura 3 - Enviar arquivos (cascata).

Figura 5 - Com Pesquisa (scrum).



Fonte: Dos autores.

Figura 6 - Sem Pesquisa (cascata).

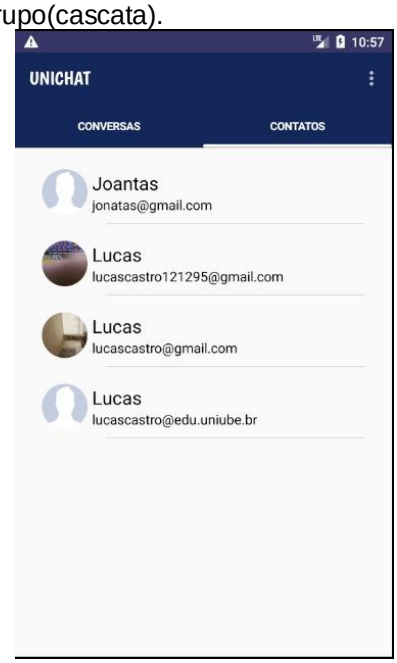


Fonte: Dos autores.

Figura 7 - Criar grupo (scrum).

Fonte: Dos autores.

Figura 8 - Sem a possibilidade de criar grupo(cascata).



Fonte: Dos autores.

As diferenças aconteceram devido à possibilidade de se desenvolver *software* em ciclos, que é uma das propriedades das metodologias ágeis.

Dessa forma, ao invés de o cliente ter recebido o aplicativo somente quando estava completo, como no Cascata, quando foi finalizada a primeira *sprint*, o resultado foi apresentado e ele

compartilhou sua sincera opinião e exigiu que fossem incluídas as outras funcionalidades.

4 Discussão

Enquanto que no modelo em Cascata a comunicação é quase opcional, quando se trata do *Scrum* a comunicação é obrigatória e ocorre por meio das reuniões diárias, reuniões de planejamento da sprint, reuniões de revisão da sprint e reuniões de retrospectiva da sprint.

Ambas as metodologias, têm suas vantagens se tratando da preocupação com a qualidade e usabilidade do *software* mas, analisando o resultado final, a Scrum permitiu a geração de um *software* de maior qualidade graças à comunicação entre os membros da equipe e a comunicação entre cliente e equipe.

A metodologia que leva vantagem em relação à qualidade da documentação do *software* é, sem dúvida, a Cascata. Nela foi preciso criar uma diversidade de diagramas para planejar o funcionamento do *software* antes de dar início ao desenvolvimento.

Em relação ao envolvimento do cliente durante o desenvolvimento do *software*, na metodologia em Cascata se faz contato com ele somente no início do projeto, quando ocorre o levantamento de requisitos e, depois disso, só é contatado no dia da entrega do *software*, o que pode ser prejudicial.

Apesar de ambas as metodologias terem suas fases de testes, utilizar Scrum tornou possível explorar o modo como se testa: além do programador, outra pessoa realizou os testes. Dessa forma, os testes não foram deixados somente na responsabilidade de quem criou o código e mais erros foram descobertos.

Quanto à facilidade de dar manutenção no *software*, em Scrum foi mais simples. De acordo com a definição de pronto criada, o código deveria ser comentado de forma construtiva. Logo, era um código auto explicativo.

Se tratando do nível de aceitação para alterações de requisitos no software, *Scrum* foi bem mais flexível. As alterações que o cliente exigiu durante a reunião foram realizadas com facilidade.

Realizar alterações no aplicativo desenvolvido em Cascata seria muito mais complexo e demorado devido ao código pouco organizado, sem citar a possibilidade de se criar erros.

5 Conclusão

A metodologia que gerou um *software* de maior qualidade foi, sem dúvidas, a *Scrum*. O desenvolvimento de projetos utilizando essa metodologia trouxe a convicção de que o produto a ser desenvolvido sempre esteve em nosso controle. As reuniões diárias deixavam todos os integrantes do grupo atentos a todos os detalhes e essa comunicação foi de suma importância para o sucesso do aplicativo desenvolvido. A forma de trabalhar no *Scrum* é um pouco complexa, mas gera efeitos significativos no decorrer do projeto.

Referências

CÁCERES, Luis. **Metodologias ágeis ou tradicionais:** quais são as melhores para o meu projeto? Disponível em: https://administradores.com.br/artigos/metodologias-ageis-ou-tradicionais-qual-e-a-melhor-para-meu-projeto#:~:text=A%20primeira%20grande%20diferença%20entre,descobrindo%20o%20percurso%20no%20caminho.. Acesso em: 01 out. 2020.

BERNARDO, Kleber. **Manifesto ágil, como tudo começou**. [*S. l.*]: Cultura Ágil, 8 dez. 2014. Disponível em: https://www.culturaagil.com.br/manifesto-agil-como-tudo-comecou/. Acesso em: 01 out. 2020.

CASA DA CONSULTORIA. **Modelo em cascata:** o que é e como funciona. Disponível em: https://casadaconsultoria.com.br/modelo-cascata/. Acesso em: 01 out. 2020.

DESENVOLVIMENTO ÁGIL. **Scrum**. Disponivel em: https://www.desenvolvimentoagil.com.br/scrum/. Acesso em: 01 out. 2020.

DIAS, Ricardo Pereira. **O modelo em cascata**. Disponível em: https://medium.com/contexto-delimitado/o-modelo-em-cascata-f2418addaf36. Acesso em: 01 out. 2020.

FONTES, Alexia. **Scrum:** entenda como funciona esse método ágil. Disponivel em: https://www.voitto.com.br/blog/artigo/scrum. Acesso em: 01 out. 2020.

SCHWABER, Ken; SUTHERLAND, Jeff. **Guia do Scrum**: Um guia definitivo para o Scrum: As regras do Jogo. 2. ed. Scrum Guides, 2017. Disponível em: https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-Portuguese-Brazilian.pdf. Acesso em: 14 out. 2020.