Desenvolvimento ágil com o método *SCRUM*

Eduardo Silva[1], Pedro Batista[2], Luís Barata[3]
[1]Instituto Politécnico de Castelo Branco, edu18082001@gmail.com
[2]Instituto Politécnico de Castelo Branco, pedrobatista2502@gmail.com
[3]Instituto Politécnico de Castelo Branco, luis.barata@ipcb.pt

Resumo

Este trabalho de investigação tem como objetivo aprofundar um método de desenvolvimento ágil muito utilizado nos dias de hoje, o *SCRUM*. O mais relevante neste artigo é saber o que é realmente o *SCRUM* e os seus valores de uma forma simples de entendimento. O *SCRUM* é uma estrutura que gere e desenvolve a entrega de um produto. O *SCRUM* é bastante útil no desenvolvimento onde os sistemas são mais complexos e incomuns. Após este trabalho de investigação, conclui-se que o *SCRUM* é um método eficiente de desenvolvimento ágil, que ajuda a gerir o desenvolvimento de projetos de forma mais eficiente.

Palavras-chave: *SCRUM*, *Framework, Sprint,* Desenvolvimento ágil

Title: Agile development with the SCRUM method

Abstract: This research work aims to deepen the theme of an agile development method used today, *SCRUM*. The most relevant thing in this article is to know what *SCRUM* really is and its values in a simple way of understanding. *SCRUM* is a framework that develops and delivers a good quality product. *SCRUM* is very useful in development where the systems are more complex and unusual. After this research work, it is concluded that SCRUM is an efficient method of agile development, which helps to manage project development more efficiently.

Keywords: *SCRUM*, *Framework*, *Sprint*, Agile Development

1. Introdução

O processo de desenvolvimento de software foi bastante desenvolvido nos últimos anos. Este processo deixou de ser feito com o recurso ao senso comum dos vários intervenientes e passou a seguir métodos e metodologias que permitiram um melhor desempenho das equipas, mas também permitiu o desenvolvimento de software, adaptado às necessidades dos utilizadores e com resultados muito positivos, em termos do processo de gestão dos recursos e funcionalidades.

Iremos, pois, dar a conhecer um pouco da história deste método, apresentando a sua origem, as suas funcionalidades e onde se aplicam, as suas vantagens e desvantagens, enquanto se pretende mostrar a sua exequibilidade com um exemplo prático onde o *SCRUM* é aplicado. Após serem descritos estes tópicos, não se pode descurar uma análise crítica sobre o tema estudado.

O *Manifesto Agile* é um conjunto de princípios e valores que foram desenvolvidos por um grupo de profissionais de software em 2001. Esse grupo reuniu-se para discutir formas de melhorar a forma como os softwares eram desenvolvidos, encontrando novas formas de aumentar a eficiência e a eficácia do processo de desenvolvimento de um software/projeto. O *Manifesto Agile* é composto por quatro valores fundamentais:

(i) Indivíduos e interações sobre os processos e ferramentas: este valor enfatiza a importância de reconhecer a importância das pessoas envolvidas no processo de desenvolvimento de software e que elas devem ter espaço para colaborar e interagir para alcançar os objetivos;
(ii) Software em funcionamento sobre a documentação completa: Este valor destaca a importância de colocar em primeiro lugar a entrega de um software que seja completamente funcional, em vez de se concentrar em produzir documentação detalhada e extensa.
(iii) Colaboração com o cliente durante a negociação de contratos: este valor enfatiza a importância de trabalhar em conjunto com os clientes para entender as suas e preferências, em vez de apenas seguir um simples contrato.
(iv) Responder a mudanças em vez de seguir um plano: Este conceito reconhece que o desenvolvimento de software é um processo dinâmico e que as mudanças podem surgir a qualquer momento, tal como o *SCRUM*. Em vez de seguir um plano rígido, é importante ser flexível e adaptar-se às mudanças que podem surgir durante o decorrer do projeto para alcançar os objetivos estabelecidos.

Existem várias metodologias ágeis, para além do *SCRUM*, cada uma com suas próprias características e abordagens, há quatro metodologias ágeis populares que são usadas nos dias de hoje:

(i) *Kanban*: O *Kanban* é uma metodologia que enfatiza o fluxo de trabalho e a melhoria contínua. Na prática, o método *Kanban* é organizado num quadro ou numa tabela, dividido em colunas, mostrando todos os fluxos dentro do projeto. Este método requer comunicação entre todos os membros e transparência, para que eles saibam o ponto de situação do projeto a qualquer momento.
(ii) *Lean*: O *Lean* é uma metodologia que se concentra em eliminar etapas desnecessárias num projeto e maximizar o valor entregue ao cliente. O *Lean* pode ser uma abordagem muito eficaz para empresas que desejam entregar valor ao cliente de maneira mais rápida e eficiente.
(iii) XP (*Extreme Programming*): XP é uma metodologia *ágile* que enfatiza o desenvolvimento de software de alta qualidade por meio de práticas como testes automatizados, programação em grupo e integração contínua. É uma metodologia que

enfatiza valores como Comunicação, Simplicidade, Feedback, Coragem e Respeito, e tem como prioridade a satisfação do cliente.
(iv) *Crystal*: É uma metodologia *ágile* que se concentra em equipas menores, com processos leves e adaptáveis. O *Crystal* pode ter equipas com muitas ou poucas pessoas e cada uma dela tem um certo nome: *Crystal Clear* (equipa de 8 pessoas), *Crystal Yello*w (equipa de 10 a 20 pessoas), *Crystal Orange* (equipa de 20 a 50 pessoas) e *Crystal Red* (equipas com 50 a 1000 pessoas). O método *Crystal* foca-se em princípios tais como Pessoas, Interações, Comunidade, Competências, Talento e Comunicação, para entregar o melhor processo de software possível.

A gestão de projetos com *SCRUM* é uma abordagem ágil para gerirem de projetos de software, que enfatizam a entrega contínua de um software funcional e a colaboração entre os membros da equipa. Por outro lado, a gestão de projetos sem *SCRUM* podem seguir uma abordagem mais tradicional, como a gestão de projetos em cascata, que se concentram em fases sequenciais do processo de desenvolvimento. Algumas das principais diferenças entre a gestão de projetos com e sem *SCRUM* incluem:

(i) Ciclos de trabalho: Com *SCRUM*, o projeto é dividido em sprints, ciclos de trabalho que duram de duas a quatro semanas. Cada sprint tem um objetivo claro e definido, e a equipa trabalha para criar um incremento de software funcional ao final de cada sprint. A gestão de projetos sem *SCRUM*, as fases do projeto são geralmente sequenciais e a equipa trabalha em cada fase antes de passar para a próxima.
(ii) Priorização de tarefas: Com *SCRUM*, as tarefas são priorizadas no *Backlog* do produto, uma lista de funcionalidades a serem desenvolvidas. Durante o *Sprint Planning*, a equipa seleciona as tarefas mais importantes do *Backlog* para trabalhar no sprint atual. Na gestão de projetos sem *SCRUM*, as tarefas são geralmente definidas no início do projeto e são executadas de acordo com um cronograma definido.
(iii) Papéis e responsabilidades: Com *SCRUM, existem* papéis definidos para cada membro da equipa, incluindo o *SCRUM Master*, *Product Owner* e a equipa de desenvolvimento. Cada papel tem responsabilidades específicas, incluindo a facilitação do processo, gerindo o *Backlog* do produto e criação do software funcional. Na gestão de projetos sem *SCRUM*, os papéis e as responsabilidades podem ser menos definidos e podem variar de projeto para projeto.
(iv) Flexibilidade: Com *SCRUM*, o processo de desenvolvimento é flexível e pode ser ajustado a qualquer momento com base no feedback do cliente ou nas mudanças no projeto. Na gestão de projetos sem *SCRUM*, o processo é geralmente mais rígido e tem mudanças significativas no projeto, que poderão ser mais difíceis de implementar.

Estas são algumas das diferenças entre a gestão de projetos com e sem *SCRUM*. Embora ambas as abordagens tenham seus prós e contras, o *SCRUM* é popular nas empresas de tecnologia pela sua capacidade de permitir a entrega contínua de software funcional e por enfatizar a colaboração entre os membros de toda a equipa envolvida.

2. Desenvolvimento do Tema

2.1. Evolução Histórica

Trata-se, de seguida, na linha do tempo, diferentes etapas na criação deste *framework* muito utilizado nos dias de hoje. Recuando uns anos atrás, em 1993, foi criado um *framework* muito utilizado no presente, cujo nome, já referido anteriormente, chamado *SCRUM*. Os seus criadores Jeff Sutherland, John Scumniotales e Jeff McKenna na Easel Corporation, (uma antiga empresa popular dos anos noventa de desenvolvimento de *software*) retiraram a ideia de um artigo da famosa revista Harvard Business Review chamado "The New New Product Development Game (1986)", onde Takeuchi e Nonaka criaram o termo *SCRUM* que comparava novas abordagens com desporto chamado Rugby. E porquê o jogo Rugby? A razão é muito simples: por se basear num jogo de equipa para ser atingida a vitória de uma equipa, ou seja, o objetivo do *SCRUM* é exatamente o mesmo, uma equipa que trabalha para dar o melhor produto com a melhor qualidade ao cliente, onde a Figura 1 mostra a comparação entre o *Rugby* e o *SCRUM*.

RUGBY	AGILE SCRUM
It is hard to see the ball, and looks like complete chaos. The score board indicates the points gained.	It is hard to understand the process being an outsider. Delivery of business value indicates the score.
The score depends on how closely the team collaborates and makes decisions	The quality of business value depends on how well the teams organize themselves
Players push the ball in the right direction by acting as a cohesive unit	The team delivers potentially shippable increments after every iteration acting as a cohesive unit
The "Scrum half" coordinates the traffic between the team players for an effective game play	Scrum Master mentors team members to empower them towards achieving common goals

Figura 1 - Comparação entre o Rugby e SCRUM; Fonte: lmadhavi.wordpress.com por Lmadhavi

2.2. Descrição do Tema

Em 1995, Jeff Sutherland e Ken Schwaber apresentaram o artigo "The *SCRUM* Development Process", o processo de desenvolvimento do *SCRUM*. Seis anos depois, mais concretamente em 2001, Sunderland lança, com os seus companheiros de trabalho, o "Manifesto Ágil". E, por fim, em 2016, forma-se o primeiro *SCRUM,* completamente escalável. A Figura 2 mostra um breve resumo da história do *SCRUM* desde o começo até ao presente deste *framework*.

Figura 2 - História do SCRUM; Fonte: www.knowledgehut.com

O mais relevante neste artigo é saber o que é realmente o *SCRUM* e os seus valores de uma forma simples de entendimento. O *SCRUM* é uma estrutura que desenvolve e entrega um produto de boa qualidade. Para que isso se concretize há uma equipa bem estruturada e dividida para que o produto criado tenha sucesso, sem esquecer que todos esses trabalhadores seguem cinco valores deste *framework*. A Figura 3 mostra como se processa o *SCRUM*:

Figura 3 - Processo SCRUM; Fonte: blog.europneumaq.com por Catarina Gomes

(i) O *Product Owner* tem a responsabilidade de estabelecer os requisitos do produto, define a data de lançamento, pode alterar as prioridades de cada *Sprint* e aceita ou rejeita o resultado de cada *Sprint*. O *Sprint* é precisamente todos os passos a percorrer para criar um certo produto e estima-se que cada um demora entre duas a quatro semanas.
(ii) O *SCRUM Master* está encarregue de garantir que todos os passos estão a ser cumpridos, colabora com todos os trabalhadores na criação do produto e tem a função de realizar reuniões diárias para rever todos os *Sprints* cumpridos e criar se for necessário.
(iii) A *SCRUM Team*, como o nome indica, é certamente a equipa que vai criar todo o produto. É composta por cinco a nove membros; escolhe os passos mais importantes de cada *Sprint,* são auto-organizados, ou seja, efetuam o trabalho da forma certa entre todos.

A Figura 4 mostra cinco principais valores do *SCRUM*:

(i) A coragem que os membros do *SCRUM* têm para realizar certos trabalhos duros da maneira certa, sem qualquer problema, ou seja o *SCRUM* valoriza a capacidade individual de cada um;
(ii) O respeito que todos os trabalhadores têm para com os outros, para que possam ser ajudados caso tenham alguma dificuldade em realizar os seus deveres, pois cada membro de cada equipa tem de estar pronto a ajudar o próximo;
(iii) O foco de cada equipa é muito importante, devido a cada *Sprint* ter um prazo, no qual cada equipa se foca num conjunto específico de objetivos a serem cumpridos;
(iv) O compromisso é um dos valores mais importantes do *SCRUM*, pois este *framework* conta com toda a equipa que está destinada a realizar este projeto para que no fim tenha o maior sucesso;
(v) A abertura, todos os trabalhadores envolvidos, incluindo os *StakeHolders*, falam abertamente sobre o projeto e os desafios durante a realização do trabalho.

Figura 4 – Valores do SCRUM; Fonte: SCRUM.org

Com o passar do tempo, diversos métodos de desenvolvimento de software que atuam no desenvolvimento de software foram evoluindo, o modelo "*Waterfall*" é um deles. Criado por Winston W. Royce, que acreditava que gerir um grande software deveria ser um processo iterativo, ao invés de ser um processo sequencial, ou seja, se não houvesse relacionamento entre todas as fases de desenvolvimento, iriam ocorrer erros. A Figura 5 demonstra os passos do modelo *Waterfall*:

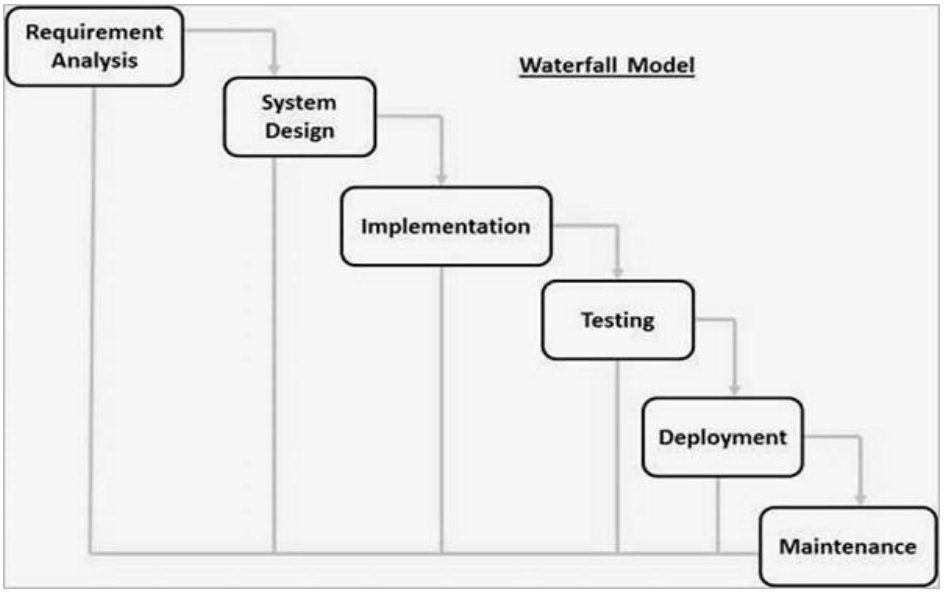

Figura 5 - Modelo Waterfall; Fonte: tutorialspont.com

(i) Análise de requisitos − Todos os requisitos possíveis do sistema que irão ser desenvolvidos são criados nesta fase e documentados num documento de especificação de requisitos;

(ii) Design do Sistema – Nesta fase, são estudadas as especificações de requisitos da primeira fase e elaborado o design do sistema. Esse design ajuda a especificar os requisitos de *hardware* e do sistema e ajuda também a definir a arquitetura geral do sistema;

(iii) Implementação − Com o projeto do sistema, o mesmo é desenvolvido numa primeira fase em pequenos programas chamados unidades, que são integrados na próxima fase. Cada unidade é desenvolvida e testada quanto à sua funcionalidade, o que é conhecido como *Unit Testing*;

(iv) Testes – Todas as unidades desenvolvidas na fase de implementação são integradas num sistema após o teste de cada unidade. Após isso, todo o sistema é testado para procurar falhas e erros;

(v) Implementação − Concluído todos testes funcionais e não funcionais, o produto é lançado no mercado ou
enviado para o cliente;

(vi) Manutenção − Poderão existir alguns problemas que surgem no cliente. Os *patches*, que são atualizações, fazem corrigir esses erros. Para melhorar o produto, como já foi dito, esses *patches* têm a competência para realizar novas atualizações.

2.3. Justificação de Aplicabilidade

A aplicabilidade do *SCRUM* permite, na equipa do projeto, um maior e mais alto nível de produtividade, contudo, como qualquer metodologia de trabalho, nem sempre o *SCRUM* é o caminho mais indicado para todos os tipos de projetos. O *SCRUM* é bastante útil no desenvolvimento onde os sistemas são mais complexos e incomuns - dada a sua estrutura, aborda o problema como um todo, mas, depois, divide-o em pequenas secções de tarefas mais simples e que podem ser executadas ao mesmo tempo. ("*SCRUM*: The Art of Doing Twice the Work in Half the Time") Além disso, em scripts e produção de blocos de código, traz uma melhor organização que permite que os StakeHolders possam ver o projeto e dar a sua opinião.

Como foi referido, nem sempre o *SCRUM* é a melhor solução, por vezes, em projetos extremamente complexos, nem todas as atividades conseguem ser identificadas e definidas de início, o que cria uma interação entre o produto final e todas os *StakeHolders* do projeto, identificando novas tarefas que seriam necessárias fazer e que não constavam nas tarefas iniciais.

Como de esperar, também os elementos da equipa vão identificar novos aspetos que anteriormente não foram considerados, ao longo da criação do projeto, com várias reuniões semanais e também com os comentários e opiniões de outros, criando-se esta adaptação aos novos requisitos de forma eficiente e rápida.

Sendo assim, o *SCRUM* é adequado quando as próprias *StakeHolders* entendem que não irão ter uma imagem completa do produto final quando a apresentam o seu software ao *SCRUM*, justificando apenas a utilização do método se os *StakeHolders* se sentirem confiantes com o *SCRUM*, perante a ausência de prazos e alterações que podem ser realizadas, mudando o conceito inicial. A adoção do *SCRUM* justifica-se também quando os programadores de uma empresa têm a capacidade de realizar tarefas de forma autónoma, o que permite ao *SCRUM* ser utilizado da forma mais eficaz, aumentando assim o nível de produtividade, dividindo as tarefas por cada programador.

Pelo contrário, cada elemento da equipa procura resolver os problemas que se desenvolvem o mais rapidamente possível, para não pôr em causa a totalidade do projeto e atrasar o mesmo. As "*Daily SCRUM"* meetings contribuem para as resoluções de problemas que surgem, pois há um acompanhamento permanente do trabalho executado por todos, entre todos. Considerando que os elementos da *Team* poderiam eventualmente prejudicar a sua performance, é aí que o papel do "*SCRUM Master"* cumpre uma tarefa excecional para o sucesso do método, pois é ele que contribui muitas vezes com uma nova perspetiva nas tarefas que estão a ser executadas. No entanto, ao adotar o *SCRUM*, também se deve estar ciente de que terá de dar-se completa autonomia à equipa e não interagir diretamente com a "*SCRUM Team*". Outro aspeto a ter em conta é a tendência do excesso de reuniões com

todos os elementos da equipa, o que acaba por retirar tempo para a realização das tarefas do projeto. Além disso, como existe autonomia da "*SCRUM Team"*, na ausência de reuniões diárias, certamente o *sprint* terminaria com incongruências entre os blocos de código desenvolvidos pelos programadores e chegar-se-ia à conclusão de que alguns desses módulos não correspondiam exatamente ao objetivo traçado no início do *sprint*, ainda que o "*SCRUM Master"* tivesse acompanhado o trabalho durante esse período.

Os erros que não se acumulam, ao longo do sprint, são resolvidos diariamente. Uma troca constante de ideias e partilha de sugestões de elementos da equipa permite melhorar o trabalho dos colegas. O contexto das "*Daily SCRUM meetings"* - e aquilo que elas implicam em si - justifica a necessidade de aplicar este método a equipas de pequena dimensão. No entanto, quando o projeto tem uma dimensão maior, com muitos mais elementos, é possível, mesmo assim, adaptar, criando uma "*SCRUM of SCRUMs*". O termo, originalmente mencionado em 2001 por Jeff Sutherland, representa o conceito de ter várias equipas *SCRUM* e uma equipa de representantes das mesmas que se reúnem com a mesma frequência das equipas *SCRUM* individuais, avaliando assim o progresso, as dificuldades e as dependências de software das equipas. O método *SCRUM* é eficiente, precisamente por ter o máximo cuidado no planeamento dos *sprints*. Quão mais aproximado o plano estiver do trabalho realmente executado, melhor a equipa conseguiu medir a complexidade das tarefas selecionadas. Se, pelo contrário, não se conseguiu concluir o que estava definido no "*Sprint Backlog*", é na "*Sprint Review*" que se deve determinar o porquê.

Finalmente, podemos também considerar uma mais-valia a existência do "*Product Owner"*, como motivo para o bom desempenho do método - o facto de apenas uma pessoa comunicar com os *StakeHolders* para definir o que é pretendido do software a desenvolver, como os requisitos iniciais que são decididos somente pelo "*Product Owner"*. Podemos verificar que o *SCRUM* é uma forma eficaz em colocar pessoas com diferentes valências a comunicar e cooperar eficientemente, com um objetivo em comum – o sucesso do projeto. Todos estes aspetos resultam num método ágil; é com base nos mesmos que se toma a decisão, na hora de escolher o processo mais adequado à realidade concreta de uma equipa/empresa e, assim, explicam a alta eficiência do *SCRUM*.

2.4. Vantagens e Desvantagens

Como já foi referido anteriormente, nem todos os métodos criados para serem usados em projetos/*softwares* são completamente eficientes e não dão problemas. O *SCRUM* tem vantagens e desvantagens ao ser usado. Após algumas pesquisas, são descritas três vantagens e três desvantagens. Começando pelas vantagens:

(i) A flexibilidade deste método é essencial para definir todos os requisitos e é capaz de dar outras novas soluções à medida que o projeto decorre;
(ii) Ao operar este processo dá a vantagem de colocar no mercado o produto, antes de o mesmo estar completamente concluído;
(iii) Os custos são mais baixos ao usar este método, devido à redução de documentos/requisitos, há uma maior produtividade devido a organização da equipa o que leva também a que o produto seja uma melhor qualidade no envio ao cliente;

Falando das desvantagens de usar o *SCRUM*:

(i) O grupo de trabalho tem de ser altamente habilidoso e treinado para implementar este método de desenvolvimento, devido a ser altamente ágil;
(ii) Devido a diversas reuniões que a equipa tem, poderá haver sempre novas ideias, o que leva a que haja sempre novos *Sprints* e o que leva a ter novos prazos;
(iii) Quando há um número acentuado de trabalhadores, há uma maior dificuldade na implementação das tarefas propostas.

3. Exemplo de Aplicação

A gestão de stock do bar da Escola Superior de Tecnologia é feita sempre no fim de cada dia de trabalho. No início de cada semana, chegam sempre produtos para serem vendidos aos alunos, professores e todos aqueles que visitam a escola. Certo dia, a funcionária responsável do bar, que recolhe os produtos fornecidos, depara-se com uma carga muito grande de comida e de bebidas de um certo fornecedor. Na semana seguinte, a funcionária recolhe novamente uma carga grande de produtos, sabendo que haveria ainda produtos suficientes para serem vendidos nessa semana. Chegou-se à conclusão de que houve um erro, ao colocar o valor correto dos produtos, ainda em stock, no bar da escola. A responsável do bar teve uma ideia para um projeto para os alunos do curso de Engenharia Informática e propô-la ao coordenador de escola.
O coordenador aceitou o projeto, indigitou um gestor para ser o "*Product Owner*" e marcou uma reunião com os alunos interessados.
Decorreu da reunião um projeto que consistiu em desenvolver uma aplicação android, que permita às funcionárias contarem o stock e, logo de seguida, procederem ao registo do valor certo quando fazem a contagem, que está ligada ao sistema que comunica ao fornecedor se é necessário enviar mais produtos para a escola. O "*Product Owner*", criou as seguintes ideias para o projeto:

1. Criação de uma nova interface para o sistema de contagem;
    a. Novo design mais recente;
    b. Verificar a segurança do sistema;
    c. Mostrar erro ao escrever um número acima do valor em stock;

2. Compra de smartphones Android;
    a. Criar a aplicação de contagem;
    b. Configurar com um U*sername* e *password*;
    c. Configurar só para uso único de contagem de stock.

Após todas estas ideias serem bem discutidas e aprovadas, o "*Product Owner*", apresenta estas soluções ao *SCRUM Master*, a um aluno, que concluiu o seu CTESP com um projeto criado em Android. Logo de seguida, esse aluno, organizou a sua equipa de 10 elementos. Juntaram-se para fazer a primeira "*Sprint Planning Meeting"* para o *"Product Owner"* mostrar, a todos os trabalhadores, a lista de tarefas que irão ter de seguir e decidir o que

realizar no primeiro *Sprint*, a que se dá o nome de *Product Backlog*. No primeiro *Sprint*, priorizou-se a criação de uma nova interface para o sistema de contagem.

1. Aproveitar a interface já existente e melhorar;
2. Criar uma interface diferente para funcionário e administrador;
3. Criar utilizadores para funcionário e administrador;
4. Rever todos os produtos e adicionar novos se for necessário;
5. Rever todos os contactos dos fornecedores;

A duração deste *Sprint* teve a duração de duas semanas; todos os dias foram realizados as "*Daily SCRUM Meetings"* com todos os trabalhadores. Em cada reunião, cada elemento tem de explicar a todos o que foi realizado, se ocorreu algum problema e também combinar quais os próximos passos para se realizar um novo *Sprint*.

1. Compra de Smartphones;
2. Criar contas para funcionárias para entrarem na aplicação de contagem de stock;
3. Configurar o smartphone para uso único de contagem de stock;

Concluindo, as Figuras 6, 7 e 8 mostram o "*SCRUM Board",* ao longo dos *Sprint*.

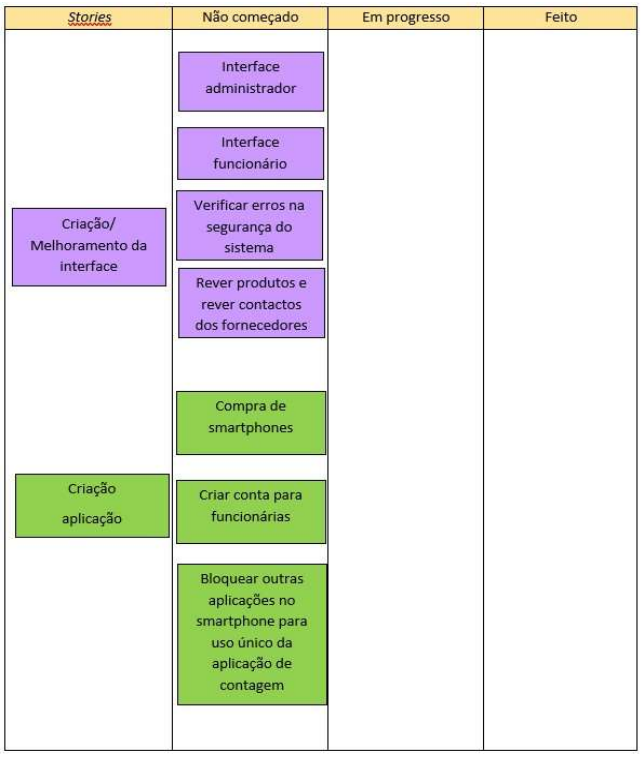

Figura 6 - Inicio do Sprint; Fonte: própria

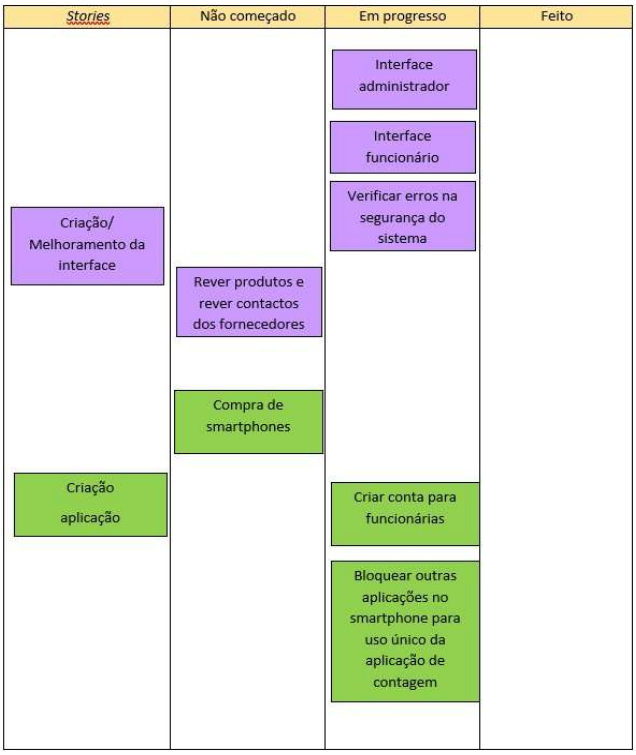

Figura 7 - Durante o Sprint; Fonte: própria

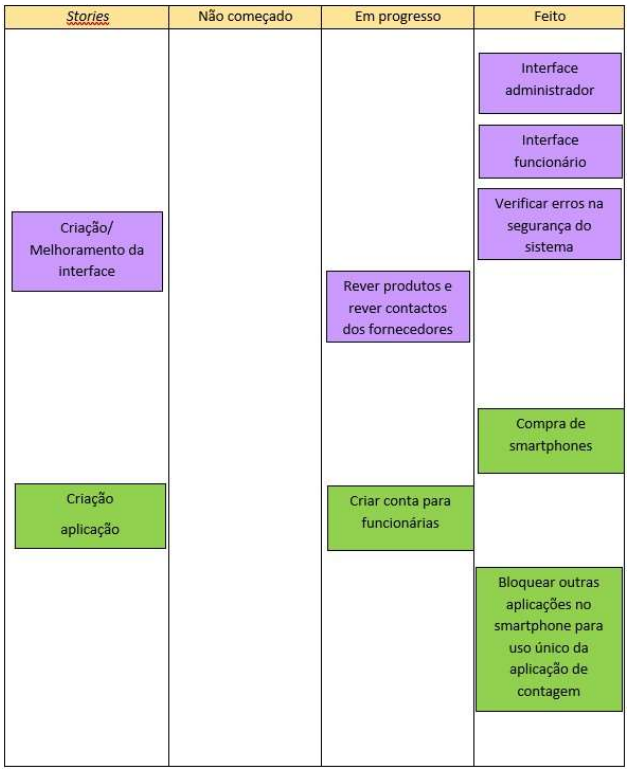

Figura 8 - Fim do Sprint; Fonte: própria

4. Análise Crítica e Conclusão

Quando se adota o *SCRUM*, é necessário que todos os elementos se comprometam com o projeto e com a equipa, isto resume a essência do método, que só tem sucesso se este compromisso existir, de facto, por todas as partes e elementos do projeto. Na realidade, *SCRUM* é um método difícil de adotar, visto as empresas terem os seus próprios princípios associados, o que faz colidir com o método *SCRUM* com as práticas das empresas. Tais hábitos como o gestor de projeto controlar o processo, atribuir tarefas e não estar, por vezes, diretamente envolvido na atividade da equipa e como também a própria equipa possa não comunicar eficazmente entre si, podem não traduzir-se, necessariamente, no bom desempenho do projeto. Todos estes hábitos são muito difíceis de quebrar. É comum, sobretudo quando um projeto avança com "*Product Owners*" e/ou "*SCRUM Master"* pouco experientes, que, inicialmente, a equipa esteja entusiasmada e confiante no sucesso da aplicação do método, mas, à medida que surgem obstáculos, a insegurança aumenta entre todos os elementos e os princípios *SCRUM* são rapidamente esquecidos. Se aliarmos estes vícios a uma administração que não confere total autonomia às equipas, a implementação deste método está condenada à partida. Assim, provavelmente, fazer perguntas de modo a aferir a mentalidade vigente e, gradualmente, introduzir e encorajar práticas mais ágeis no dia-a-dia de todos, antes de se decidir se o *SCRUM* pode/deve ser implementado, é mais importante estudar o comportamento da empresa em que se pretende inserir o método do que simplesmente querer usar o programa em si. A médio prazo, será possível apreciar a criação de novos hábitos mais vincados, que devem ser realçados, para que todos possam interiorizar as mais-valias da atitude ágil. Após este trabalho de investigação, conclui-se que o *SCRUM* é um método eficiente de desenvolvimento ágil, que ajuda a gerir o desenvolvimento de projetos de forma mais eficiente. Ao usar este método, mostra-se todo o progresso do produto, permitindo a todas as equipas de desenvolvimento terem uma maior facilidade em se focar numa certa tarefa, devido à criação dos *Sprints* para um certo produto, o que faz com que eles estejam mais organizados. Apesar de ter certas limitações, sendo a principal desvantagem a de o trabalhador ter de ser altamente habilidoso para conseguir aplicar os seus conhecimentos, não deixa de ser um processo hábil.

Bibliografia

Agilest LLC, "*Agile SCRUM Methodology*", [Online]. Acedido a 2 de outubro de 2022, em: https://www.agilest.org/SCRUM/

Agilest LLC, "*Daily SCRUM Meeting*", [Online]. Acedido a 2 de outubro de 2022, em: https://www.agilest.org/SCRUM/daily-SCRUM-meeting/

Agilest LLC, "*Sprint Retrospective Meeting*", [Online]. Acedido a 2 de outubro de 2022, em: https://www.agilest.org/SCRUM/sprint-retrospective/

Agile Alliance, "*SCRUM of SCRUMs*", [Online]. Acedido a 30 de setembro de 2022, em: https://www.agilealliance.org/glossary/SCRUM-ofSCRUMs/#q=~(infinite~false~filters~(postType~(~'page

~'post~'aa_book~'aa_event_session~'aa_experience_report~'aa_glossary~'aa_research_paper~'aa_video)~tags~(~'SCRUM*20of*20SCRUMs))~searchTerm~'~sort~fals e~sortDirection~'asc~page~1)

Ajmal, S. (26 de maio de 2022). "*Pros and Cons of SCRUM Methodology*", [Online]. Acedido a 2 de outubro de 2022, em: https://www.quickstart.com/blog/pros-andcons-of-SCRUM-methodology/

Cobb, C. "*What Are the Advantages and Disadvantages of Agile and SCRUM?*", [Online]. Acedido a 1 de outubro de 2022, em: https://managedagile.com/what-are-theadvantages-and-disadvantages-of-agile-SCRUM/

Contezini, D. (1 de março de 2017). "*Descubra quais são as vantagens e desvantagens do SCRUM*", [Online]. Acedido a 1 de outubro de 2022, em: https://blog.asaas.com/4-vantagens-e-desvantagensdo-SCRUM-para-negocios-focados-em-saas/

Cho, J. J, (2010). An Exploratory Study on Issues and Challenges of Agile Software Development with SCRUM. *All Graduate Theses and Dissertations.* 599.

Drumond, C. "*SCRUM: Learn how to SCRUM with the best of'em"*, [Online]. Acedido a 1 de outubro de 2022, em: https://www.atlassian.com/agile/SCRUM

Gomes, C. (26 de abril de 2017). "*SCRUM: A Metodologia Ágil Simplificada*", [Online]. Acedido a 3 de outubro de 2022, em https://blog.europneumaq.com/SCRUM-metodologiaagil-simplificada

Guthrie, G. (2022). "*What are the 5 SCRUM values, and why are they important?*", [Online]. Acedido a 1 de outubro de 2022, em: https://nulab.com/learn/projectmanagement/what-are-the-5-SCRUM-values-and-whyare-they-important/

Hossain, E., Babar, M. A., & Paik, H.-y. (2009). Using SCRUM in Global Software Development: A Systematic Literature Review. *Fourth IEEE International Conference on Global Software Engineering*, pp.175184.

Knowledgehut. "*SCRUM History*", [Online]. Acedido a 30 de setembro de 2022, em: https://www.knowledgehut.com/tutorials/SCRUMtutorial/SCRUM-history

Leite, L. M., & Lucrédio, D. (2014). Desenvolvimento de Software utilizando o Framework *SCRUM*: um Estudo de Caso. *Departamento de Computação - Universidade Federal de São Carlos, SP, Brasil,* pp. 114-121.

Lmadhavi, (11 de abril de 2018). "*History of SCRUM*", [Online]. Acedido a 30 de setembro de 2022, em: https://lmadhavi.wordpress.com/2018/04/11/historyof-SCRUM/

Pereira, P., Torreão, P., & Marcal, A. S. (2007). Entendendo *SCRUM* para Gerenciar Projetos de Forma Ágil. *MundoPM*, pp. 1-11.

Prasetya, K. D., Suharjito, & Pratama, D. (2021). Effectiveness Analysis of Distributed SCRUM Model Compared to Waterfall approach in Third-Party Application Development. *Procedia Computer Science 179*, pp. 103-111

ProductPlan.com, "*Agile Values*", [Online]. Acedido a 29 de março de 2023, em: https://www.productplan.com/glossary/agile-values/

Santo, D. E., (2022). "*Top 5 main Agile methodologies: advantages and disadvantages*", [Online]. Acedido a 29 de março de 2023, em: https://www.xpand-it.com/blog/top-5-agile-methodologies/

Schwaber, K., & Sutherland, J. (2020). "*The SCRUM Guide"*, [Online]. Acedido a 30 de setembro de 2022, em: https://billlewistraining.com/wp-content/uploads/2017/02/PMP-Agile-StudyMaterials.pdf

Silva, F., & Monforte, L. (outubro 2019). *Desenvolvimento Ágil SCRUM*. Trabalho (Licenciatura em Engenharia Informática) – Instituto Politécnico de Castelo Branco. Castelo Branco, pp. 1-23.

Soares, M. S. (2004). Metodologias Ágeis Extreme Programming e SCRUM para o Desenvolvimento de Software. *Universidade Presidente Antônio Carlos, BR, Brasil.*

SCRUM.org, *"What is SCRUM?: A Better Way Of Building Products*", [Online]. Acedido a 2 de outubro de 2022, em: https://www.SCRUM.org/resources/what-isSCRUM?gclid=CjwKCAjwqJSaBhBUEiwAg5W9p0KLnNMxLV4dGfPQXp1Wa0oTBMHG4NbnI6xeRoMZ9IN13p1hynAuLBoCe2UQAvD_BwE

SCRUM.org, "*What is a SCRUM Master?*", [Online]. Acedido a 2 de outubro de 2022, em: https://www.SCRUM.org/resources/what-is-a-SCRUMmaster

SCRUM.org, "*What is a Daily SCRUM?*", [Online]. Acedido a 2 de outubro de 2022, em: https://www.SCRUM.org/resources/what-is-a-dailySCRUM

SCRUM.org, "*What is a Sprint Review?*", [Online]. Acedido a 2 de outubro de 2022, em: https://www.SCRUM.org/resources/what-is-a-sprintreview

SCRUM.org, "*What is a Product Owner?*", [Online]. Acedido a 2 de outubro de 2022, em: https://www.SCRUM.org/resources/what-is-a-productowner

Sutherland, J. (2014). *The Art of Doing Twice the Work in Half the Time*. New York: Random House

Takeuchi, H., & Nonaka, I. (1986). The new new product development game. *Havard Business Review*, pp. 137-146.

Tutorialspoint. "*SDLC – Waterfall Model*", [Online]. Acedido a 3 de outubro de 2022, em: https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm

Xebrio.com, "*PRiSM Project Management Methodology*", [Online]. Acedido a 29 de março de 2023, em: https://www.xebrio.com/project-management-methodology/prism-project-management/

Eduardo R. Silva nasceu em Coimbra, Portugal, em 2001, residente em Castelo Branco. Concluiu o 12.º Ano na Escola Secundária Amato Lusitano em Castelo Branco e apresentou como Projeto de Aptidão Profissional, um software de gestão de reservas de um restaurante. Técnico Superior de Redes e Sistemas Informáticos pela Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco com um estágio final de curso realizado na APTIVPORT Services, S.A em Castelo Branco. Frequenta atualmente o curso de Licenciatura de Engenharia Informática, no qual ingressou em 2021.

Pedro Batista, nasceu em Castelo Branco, Portugal, em 2001. Concluiu o 12° Ano em Gestão e Programação de Sistemas Informáticos na Escola Secundária Amato Lusitano em Castelo Branco e apresentou como Projeto de Aptidão Profissional, um software de controlo de música numa coluna de som. Técnico Superior de Redes e Sistemas Informáticos na Escola Superior de Tecnologia no Instituto Politécnico de Castelo Branco com estágio final realizado na Alarme On - Soluções de Segurança em Castelo Branco. Frequenta atualmente o curso de Licenciatura de Engenharia Informática, no qual ingressou em 2021.

Luís M. Barata nasceu em Castelo Branco, Portugal, em 1977. Licenciado em Engenharia Informática pela Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco. Mestre em Desenvolvimento de Software e Sistemas Interativos na mesma instituição. Atualmente a realizar um doutoramento em Engenharia Informática no Departamento de Informática da Universidade da Beira Interior, na Covilhã. Colabora, desde 2014, com a Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco, atualmente, como professor adjunto convidado no curso de Licenciatura de Engenharia Informática.