UNIVERSIDADE FEDERAL DE JUIZ DE FORA
CURSO DE GRADUAÇÃO DE ENGENHARIA DE PRODUÇÃO

LUCAS FRANCO FERREIRA DA SILVA

METODOLOGIAS ÁGEIS EM PROJETOS DE DESENVOLVIMENTO DE SOFTWARE

Juiz de Fora
2022

LUCAS FRANCO FERREIRA DA SILVA

METODOLOGIAS ÁGEIS EM PROJETOS DE DESENVOLVIMENTO DE SOFTWARE

Trabalho de Conclusão de Curso apresentado à Universidade Federal de Juiz de Fora como requisito parcial para obtenção do título de Engenheiro de Produção.

Orientador: Prof. Dr. Marcos Vinícius Rodrigues

Juiz de Fora

2022

Ficha catalográfica elaborada através do Modelo Latex do CDC da UFJF com os dados fornecidos pelo(a) autor(a)

SILVA, LUCAS F. F..
METODOLOGIAS ÁGEIS EM PROJETOS DE DESENVOLVIMENTO DE SOFTWARE / LUCAS FRANCO FERREIRA DA SILVA. – 2022.
58 f. : il.

Orientador: Prof. Dr. Marcos Vinícius Rodrigues
Trabalho de Conclusão de Curso – UNIVERSIDADE FEDERAL DE JUIZ DE FORA, CURSO DE GRADUAÇÃO DE ENGENHARIA DE PRODUÇÃO. , 2022.

1. Desenvolvimento de *Software*. Metodologias Ágeis. Scrum. Kanban. Lean. Rodrigues, Marcos V., orient.

LUCAS FRANCO FERREIRA DA SILVA

METODOLOGIAS ÁGEIS EM PROJETOS DE DESENVOLVIMENTO DE SOFTWARE

Trabalho de Conclusão de Curso apresentado a Faculdade de Engenharia da Universidade Federal de Juiz de Fora, como requisito parcial para a obtenção do título de Engenheiro de Produção.

Aprovada em 09 de Agosto de 2022.

BANCA EXAMINADORA

D. Sc. Marcos Vinícius Rodrigues (Orientador)
Universidade Federal de Juiz de Fora

D. Sc. Roberta Cavalcanti Pereira Nunes
Universidade Federal de Juiz de Fora

D. Sc. Antonio Angelo Missiaggia Picorone
Universidade Federal de Juiz de Fora

AGRADECIMENTOS

Agradeço a Deus pela oportunidade de concluir a realização deste sonho. Aos meus pais, Jussara e Darcy, a minha namorada Marcela, e à minha família, que sempre foram fonte de inspiração e me apoiaram e incentivaram em todos os momentos.

Agradeço ao corpo de professores do curso, que dedicam sua vida a compartilhar conhecimento e formar profissionais capazes a contribuir na melhoria do mundo.

Agradeço em especial ao meu orientador Marcos, por todo apoio e incentivo no desenvolvimento deste trabalho, bem como disponibilidade para acomodar os encontros mesmo com uma agenda apertada.

Agradeço também a todas empresas que já trabalhei mas especialmente a Wabtec, por me abrir as portas da engenharia a nível global, e manter acesa a chama dos valores e relevância do papel um engenheiro na sociedade.

RESUMO

O presente trabalho tem objetivo de introduzir o tema Metodologias Ágeis aplicadas ao desenvolvimento de *softwares*. A criação de programas computacionais como meio de solucionar e otimizar problemas atuais recebe cada vez mais atenção no mercado de produtos, uma vez que *softwares* suportam a expansão do setor da tecnologia e, com a existência de clientes que desejam acessar soluções mais rápidas e inovadoras. Por isso, desenvolver esse produto fez com que metodologias de gerenciamento fossem adaptadas para atender às novas necessidades do mercado do século XXI. A proposta dos Métodos Ágeis é, em resumo, fazer com que o produto seja entregue de forma mais rápida para o cliente e permite que os processos de refino e ajuste ocorram de maneira simultânea, junto do consumidor. Há, ainda, foco no cliente durante o projeto - que agora participa integralmente da criação - e na otimização e melhoria dos times de trabalho através da flexibilidade, comunicação, adaptabilidade e colaboração. Neste contexto, este estudo apresenta alguns dos métodos clássicos de gerenciamento de projetos de sistemas digitais (Cascata, Prototipação e *Rapid Application Development*) a fim de trazer uma base de entendimento ao leitor, e, ainda, traz conceitos sobre as metodologias ágeis. Não obstante, um estudo de caso é analisado como modo de entender as nuances práticas envoltas na utilização do Scrum para desenvolvimento de sistemas em uma empresa de pequeno porte. A aplicação do método apresentou resultado eficazes e os resultados da pesquisa aplicada mostraram que a equipe envolvida no projeto alegou melhora na comunicação, entregas mais direcionadas e ambiente, de modo geral, mais propício para seu desenvolvimento. O grau de satisfação do cliente e o retorno financeiro não puderam ser medidos, pois o *software* não foi comercializado. Entretanto, de modo geral os resultados foram bastante positivos. Não obstante, deve-se considerar o ambiente de aplicação dos métodos ágeis e toda a estrutura necessária para seu emprego eficaz. Deve existir uma cultura ágil, com pessoas treinadas e todo um ambiente propício para a aplicação do método de maneira real e eficaz. Esses pontos serão discutidos em mais detalhes ao longo do presente trabalho.

Palavras-chave: Metodologias Ágeis. Desenvolvimento de *Software*. Lean. Scrum. Kanban.

ABSTRACT

This study aims to introduce the topic of Agile Methodologies applied to software development. The creation of computer programs as a means of solving and optimizing current problems receives attention of technology companies, since software supports the expansion of this sector and customers want to access faster and more innovative solutions. Therefore, to develop computer systems management methodologies were adapted to meet the new needs of the 21st century market. The proposal of Agile Methods is, in short, to make the product's delivery more quickly to the customer and allows the refining and adjustment processes to occur simultaneously, with the consumer. There is also a focus on the client during the project - who now participates fully in the creation process - and on the optimization and improvement of work teams through flexibility, communication, adaptability, and collaboration. In this context, this study presents some of the classic methods of project management of digital systems (Waterfall, Prototyping and Rapid Application Development) to bring a base of understanding to the reader, and brings concepts about agile methodologies. Nevertheless, a case study is analyzed to understand the practical nuances involved in the use of Scrum for systems development. The application of the method showed effective results and the results of the applied research showed that the team involved in the project claimed improved communication, more targeted deliveries, and an environment, in general, more conducive to its development. The degree of customer satisfaction and financial return could not be measured, as the software was not commercialized. However, in general the results were quite positive. The professional one must consider the environment in which agile methods are applied and the entire structure needs to be adapted for its effective use. There must be an agile culture, with trained people and an environment conducive to the application of the method in a real and effective way. These points will be discussed in more detail throughout this work.

Keywords: Agile Methodologies. Software development. Lean. Scrum. Kanban.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

TI	Tecnologia da Informação
RAD	*Rapid Application Development*
PDP	Processo de Desenvolvimento de Produto
TPS	Toyota Production System
PDCA	*Plan, Do, Check, Act*
XP	*eXtreme Programming*
PMBOK	*Project Management Body of Knowledge*
HTML	*HyperText Markup Language*

SUMÁRIO

1 INTRODUÇÃO

1.1 CONSIDERAÇÕES INICIAIS

Após as Primeira e Segunda Revoluções Industriais entre os séculos XVIII e XIX, o mundo enfrentou intensas mudanças nos processos produtivos, bem como nos sistemas econômicos, políticos e culturais existentes. Diversas implicações trazidas pelas Revoluções foram responsáveis por alterar o cenário global em diferentes momentos, acumulando muitos desafios para as gerações futuras. Dentre tais implicações, ressaltam-se os avanços no setor de tecnologia que, principalmente durante a a Terceira Revolução Industrial no século XX, recebeu aportes de investimentos por possibilitar vantagens competitivas para o cenário da época, onde o avanço tecnológico foi essencial para definições do cenário político-econômico vigente.

Paulatinamente, algumas tecnologias impulsionadas no século XX - como, por exemplo, painéis solares e computadores portáteis - começaram a ser acessadas pela população e o mercado consumidor de itens eletrônicos cresceu muito e, atualmente, é possível notar que os principais centros urbanos tornaram-se dependentes de ferramentas como computadores, *smartphones*, aplicativos e sistemas operacionais automatizados.

O crescimento de ambos os mercados fomentou avanços nos processos produtivos e o perfil do consumidor atual, que, de modo geral, anseia por itens cada vez mais personalizados e entregues da maneira rápida, atraiu o uso de tecnologias abarcadas aos processos produtivos de diferentes mercados, até mesmo indústrias de base como setor cirúrgico e área educacional (QUINTINO et al., s.d.), a fim de trazer na transição de todos seus processos para adaptá-los à nova dinâmica.

Nesse cenário, a aplicação de *softwares* fabricados para resolver problemas com agilidade é de grande relevância, pois permite que diversos setores e processos aumentem o ritmo de suas entregas, reduzindo possíveis erros humanos através da automatização dos processos - o que gera maior confiabilidade e potencializa os resultados das corporações.

Junto ao setor de desenvolvimento de programas digitais, diversas áreas de gerenciamento de projetos têm alterado a forma como trabalham para tornar as corporações mais ágeis, competitivas e, consequentemente, com maior potencial para crescimento. Metodologias alternativas de gerenciamento têm sido aplicadas em diversos ramos, e é nesse contexto que os chamados Métodos Ágeis se destacaram (AMBLER, 2002).

Por se tratar de um produto muito particular, um sistema requer um gerenciamento peculiar em seu desenvolvimento, de forma que métodos tradicionais têm perdido cada vez mais mercado quando se trata de gerenciar produtos com aplicabilidade para o mundo digital. Por tudo isso, os Métodos Ágeis se apresentam como potenciais solucionadores dos problemas encontrados durante os projetos.

À título de curiosidade, há atores, como Vargas (2016), que acreditam existir uma

disputa entre duas vertentes de gerenciamento de projetos, onde a primeira se baseia no *Project Management Body of Knowledge* (PMBOK) - um famoso manual de boas práticas que concatena diretrizes e definições para gestão de projetos a fim de guiar as fases de orçamento, compras, cronogramas e outras competências pertinentes - e a segunda se baseia no agilismo. Os objetivos são os mesmos para ambas: finalizar projetos de criação de sistemas com a qualidade, prazo e recursos definidos. Porém, o caminho que cada frente irá adotar é particular e o obstáculo é analisar qual metodologia deve ser escolhida, ajustá-las e uni-las para entregar o produto com excelência (VARGAS, 2016).

Os modelos tradicionais de gerenciamento ainda são amplamente empregados, mas seus fundamentos, meios de controle, estilos de gerenciamento e gestão de conhecimento, papéis e responsabilidades, meios de comunicação, ciclos do projeto, formas de desenvolver e estrutura organizacional são diferentes dos métodos ágeis - conforme Figura 1, que traz os principais pontos de diferença que são citados ao longo do trabalho.

Figura 1 – Comparativo entre Métodos Ágeis e Tradicionais

	TRADICIONAL	METODOLOGIAS ÁGEIS
Pressupostos fundamentais	Sistemas totalmente especificáveis, previsíveis; desenvolvidos a partir de um planejamento extensivo e meticuloso	Software adaptativo e de alta qualidade; pode ser desenvolvido por equipes pequenas utilizando os princípios da melhoria contínua do projeto e testes orientados a rápida resposta a mudanças
Controle	Orientado a processos	Orientado a pessoas
Estilo de gerenciamento	Comandar e controlar	Liderar e colaborar
Gestão do conhecimento	Explícito	Tácito
Atribuição de papéis	Individual – favorece a especialização	Times auto-organizáveis – favorece a troca de papéis
Comunicação	Formal	Informal
Ciclo do projeto	Guiado por tarefas ou atividades	Guiado por funcionalidades do produto
Modelo de desenvolvimento	Modelo de ciclo de vida (Cascata, Espiral, ou alguma variação)	Modelo iterativo e incremental de entregas
Forma/estrutura organizacional desejada	Mecânica (burocrática com muita formalização)	Orgânica (flexível e com incentivos a participação e cooperação social)

Fonte: (PRIKLADNICKI; WILLI; MILANI, 2014)

1.2 MOTIVAÇÕES

Muito se tem falado sobre as metodologias ágeis nos dias atuais e as promessas de melhorias dos resultados qualitativos mediante sua aplicação são diversas e uma das motivações para o presente estudo consistiu em entender a visão geral de tais métodos e como se comparam com os meios de gestão tradicionais.

Um ponto impulsionador também foi o conteúdo contido no chamado Relatório CHAOS (2015), que trouxe resultados intrigantes sobre a aplicação de métodos ágeis nos últimos anos a fim de realizar uma comparação entre os resultados de projetos ágeis e projetos tradicionais em Cascata (do inglês, *Waterfall*) e, em todos os tamanhos de projetos, o agilismo obteve maiores porcentagens de sucesso e menores falhas definitivas. Dados do CHAOS são apresentados na seção 2.

Outro motivador foi a conclusão apresentada por Sawyer e Guinan (1998) no artigo chamado *Software development: Processes and performance* (1998), onde os autores destacam que o uso de métodos de produção, como metodologias tradicionais para desenvolvimento de *software* e ferramentas de desenvolvimento automatizado, não explicam a variância da qualidade dos programas desenvolvidos ou o desempenho do time. Em contra partida, processos sociais como hierarquia horizontal (ou coordenação informal), comunicação, habilidades para resolver conflitos e o nível de apoio existente entre o time podem representar até 25% da qualidade da produção do sistema (SAWYER; GUINAN, 1998).

Esse cenário associado ao contexto global de expansão dos mercados de tecnologia motivaram a elaboração do atual trabalho. O autor entende como indispensável aos alunos dos cursos de Engenharia de Produção, bem como outros interessados no assunto, conhecer, ainda que de maneira breve, as metodologias tradicionais mais encontradas no mercado de desenvolvimento de produtos digitais e também conhecer os Métodos Ágeis que têm recebido destaque no setor da tecnologia e que tem como foco as iterações entre pessoas, e não somente em processos.

1.3 OBJETIVOS

Tem-se como objetivo geral apresentar uma visão geral qualitativa de algumas das Metodologias Ágeis aplicadas ao desenvolvimento de *softwares* no mercado atual, possibilitando leitores o entendimento do contexto do agilismo no ramo da tecnologia.

Como objetivos específicos, o autor deseja:

- Apresentar ao longo do texto os autores de referência científica para referenciar os conceitos à respeito dos métodos tradicionais de desenvolvimento de produtos digitais;
- Mostrar o Manifesto Ágil e elencar seus valores e princípios;

- Comparar quesitos básicos entre os métodos tradicionais e ágeis de gestão de projetos de criação de *softwares*;
- Apresentar um estudo de caso da aplicação do Scrum em uma empresa de tecnologia e elencar os resultados obtidos pela pesquisa.

1.4 METODOLOGIA

Foram reunidas informações básicas sobre os processos que baseiam o gerenciamento de projetos de tecnologia, limitando-se a esclarecê-los de forma sucinta. O trabalho se propõe a ser fonte de informações introdutórias ao público interessado no tema, sejam profissionais, acadêmicos ou até mesmo entusiastas do assunto, e pode ser um ponto de partida para entender de maneira geral algumas das metodologias de desenvolvimento de *softwares* e também para conhecer alguns dos principais autores do tema.

Foram realizadas leituras sobre tópicos base, com referências em autores relevantes e, majoritariamente, com grande volume de citações em trabalhos, selecionados de acordo com a plataforma *Google Scholar*. Portanto, com objetivo de trazer uma leitura fluida e prazerosa, a revisão bibliográfica é apresentada ao longo do texto para corroborar informações e provar as fontes científicas aqui mencionadas. Assim sendo, a metodologia consistiu em uma pesquisa exploratória com embasamento teórico nas referências bibliográficas apresentadas de maneira fluida ao longo do texto, apenas para trazer respaldo científico às informações - e somente isso. Além disso, o estudo de caso analisado traz um exemplo de aplicação pratica do Scrum em um ambiente empresarial propício.

Deve-se mencionar que é muito comum encontrar termos no idioma inglês quando o assunto é Métodos Ágeis, de forma que várias nomenclaturas são conhecidas popularmente por seu nome nesse idioma. O autor prezou por traduzir os termos por ele entendidos como cabíveis, que apresentam aplicação relativamente popular no idioma Português. Entretanto, substantivos muito populares, como de papéis dos envolvidos nas disciplinas ou ferramentas práticas, tiveram nomenclatura preservada como forma de trazer à luz do entendimento termos que podem ser encontrados nesse mercado. Por isso, este documento pode se apresentar, ainda, como possível suporte para conhecer palavras que derivam do inglês e são comumente encontradas no setor de gerenciamento em tecnologia.

1.5 ESTRUTURAÇÃO

Este trabalho é composto por 5 capítulos, os quais estão descritos na sequência.

O capítulo 1 possui caráter introdutório, onde o tema é contextualizado e são descritas as motivações e os objetivos do estudo desenvolvido.

O capítulo 2 conta com uma revisão bibliográfica geral de alguns modelos tradicionais utilizados para criar produtos digitais aplicados na indústria de *softwares*; apresenta-se o Manifesto Ágil e são abordadas metodologias de destaque no cenário de desenvolvimento de produtos digitais.

O capítulo 3 discute as diversas formas e expericências da aplicação do Scrum.

O capítulo 4 apresenta as conclusões obtidas no estudo.

Por fim, o capítulo 5 apresenta as considerações finais e elenca algumas sugestões para trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

A forma de consumir no século XXI mudou junto com o avanço da tecnologia. Para Sipper e Bulfin (1997), a sofisticação dos consumidores é uma das causas da revolução no mercado, visto que tem sido buscadas cada vez mais a pontualidade, variedade e alta qualidade dos produtos - associados à baixos custos. Além disso, Starr (1998) coloca a flexibilidade como um objetivo que empresas devem ter para sobreviver e ter sucesso nesse novo ambiente. Pessoas e corporações têm buscado por produtos capazes de atender suas demandas e resolver problemas de maneira cada vez mais precisa, rápida e dinâmica, o que deixa evidente a oportunidade de mudanças nos processos produtivos como meio de aumentar a competitividade de corporações no mercado (BROWN; EISENHARDT, 1995). Diante desse cenário e do notável crescimento do setor de tecnologias, no século XXI surge a Indústria 4.0 e a tecnologia passa a ser empregada nas mais diversas soluções de mercado com objetivo de otimizar as entregas, revolucionar os processos produtivos e aumentar a competitividade entre corporações.

Fica evidente como a ciência é, desde o século XVIII, um diferencial para o crescimento dos mercados econômicos e como desempenha papel transformador no mundo. Em 2021, a consultoria global chamada Bain & Company realizou um estudo de nome *Technology Report* para mostrar como corporações que trazem inovações como o principal foco de sua estratégia, sejam elas do setor de tecnologia, também chamado de *tech* (do inglês, *technology*) ou não, são as que mais expandiram seu valor de mercado na época e, ainda de acordo com o Estudo, que entre os anos de 2015 a 2020, o setor de consumo foi o segundo com maior crescimento (COMPANY, 2021).

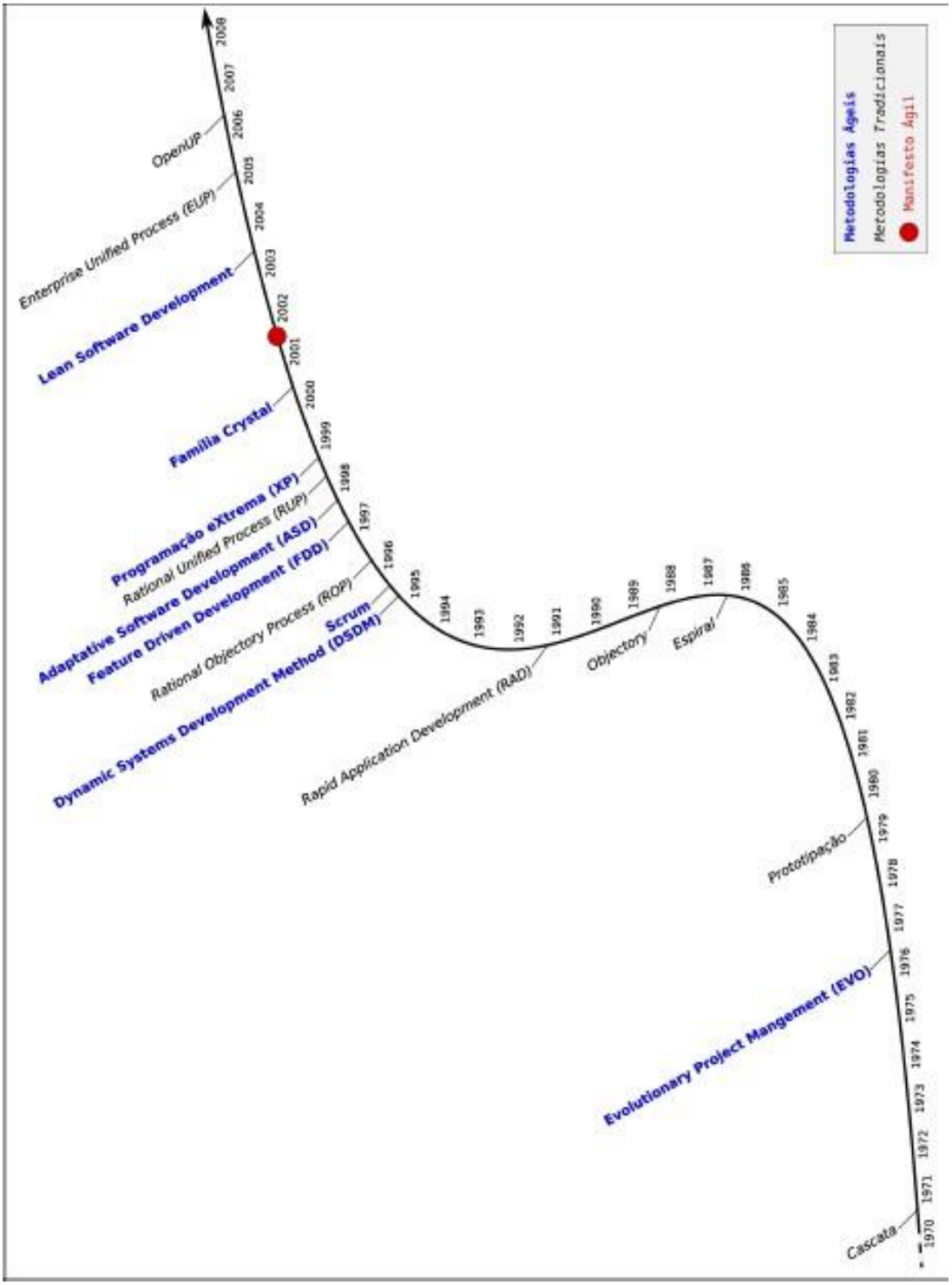
Entre as décadas de 60 e 70, os meios para fazer produtos digitais, também chamados de modelos de ciclo de vida, eram desorganizados e sem estruturação adequada, ou planejamento, para gerar o produto final e isso resultou em produtos com qualidade baixa e, ainda, sem registro documental de suas funcionalidades. Como a reincidência de projetos falhos era alta, esse período ficou conhecido como Crise de *Software* (PRESSMAN; MAXIM, 2021).

A Crise trouxe à tona a necessidade de padronizar o modo de criação de produtos digitais a fim de aumentar a qualidade do produto final, para atender às necessidades dos clientes de maneira adequada. Surgem, então, os métodos clássicos de desenvolvimento de sistemas, também chamados de modelos orientados à objetos devido ao volume de documentação exigida nas etapas dos projetos (BOEHM, 1988).

Em contraste aos modelos tradicionais de gestão, o Manifesto Ágil foi criado em um encontro em Utah no ano de 2001 por 17 personagens da cena de tecnologia de *softwares* e soluções digitais estadunidense. O Manifesto emergiu na cena de gerenciamento de projetos e trouxe mudanças na forma de fazer esses produtos, ressaltando a importância da valorização dos indivíduos e interações e a relevância de concretizar entregas significativas para o cliente, para que a solução desejada seja alcançada da maneira mais rápida possível e de forma coorperativa, e que também exista uma resposta dinâmica aos imprevistos e mudanças de prioridades. São

propostas modos diferenciados de gestão voltada para o desenvolvimento de *softwares*, onde metodologias como o Scrum, Kanban, Lean , *eXtreme Programming* (XP) e outras, têm sido adotadas por empresas dentro, não somente do setor tecnológico, mas também da Indústria 4.0. Essas novas metodologias que permeiam o mercado de corporações, indústrias e academias no atual contexto mundial. O histórico de publicações sobre as metodologias para gerenciamento de projetos de sistemas digitais pode ser verificado na Figura 2.

Figura 2 – Histórico de publicações de metodologias de desenvolvimento de softwares



Fonte: (BASSI FILHO, 2008)

O instituto americano de pesquisa chamado The Standish Group, que atua no ramo de Tecnologia da Informação (TI) desde 1985 e desenvolve pesquisas sobre fatores de sucesso e fracasso de projetos de *softwares* desde 1985, elaborou o chamado *CHAOS Report*. O estudo feito em duas edições definitivas em 1994 e em 2021 e com outras publicações de atualização entre esses anos, traz análises sobre os fatores que levam projetos que utilizam Métodos Ágeis - comumente projetos de tecnologia - ao sucesso ou fracasso quando comparados aos métodos tradicionais de gestão de projetos, como os utilizados para construção de pontes.

A premissa para a analogia entre pontes e *softwares*: sem generalizações, projetos de pontes têm orçamento definido, são construídas no prazo e não caem. Em contra partida, um programa digital nunca é entregue no prazo, ou possui orçamento definido, além de sempre apresentar falhas (CLANCY, 1995). Ainda de acordo com o grupo Standish, projetos de pontes recebem um nível extremo detalhe e por isso sua taxa de falha é muito pequena, por isso tratam-se de trabalhos engessados, com pouco dinamismo ou flexibilidade frente à imprevistos ou mudanças; são projetos onde os gerentes investem seu tempo em prever as possíveis falhas e imprevistos para evitar ter que lidar com os mesmos quando surgem. Entretanto, no ambiente dinâmico e cada vez mais rápido da revolução atual, metodologias congeladas de gerenciamento não acomodam mudanças nas práticas de negócios (CLANCY, 1995) e modelos mais flexíveis devem ser adotados - a ausência dessa flexibilidade pode ser uma justificativa para o fracasso nos projetos de *softwares*.

Em 2018, o Grupo propôs uma análise para avaliação do sucesso de um projeto para algo que vai além da tríade prazo, custo e objetivo do escopo com a justificativa de que esses pilares definem o sucesso da gestão do projeto, mas não esclarecem quanto à qualidade do produto. Por isso, as análises do CHAOS começaram a incorporar a satisfação dos clientes mediante a entrega final do *software*. A primeira edição do CHAOS considera três fatores de classificação para os projetos analisados - prazo, custo e escopo; já em 2015, foi incorporado esse índice de satisfação do cliente (STANDISH GROUP, 2015).

Como ilustra a Figura 3, de acordo com o Relatório CHAOS (2015), para Projetos de Grande Porte a taxa de sucesso utilizando metodologias ágeis é de 18%; para Projetos de Médio Porte (é de 27%; e 58% para Projetos de Pequeno Porte. Enquanto para criações de mesmo porte e que utilizaram a metodologia tradicional em Cascata, as médias foram de 3%, 7% e 44%, respectivamente.

A definição de Sucesso foi atrelada a seis atributos individuais: Prazo, Custo, Meta, Valor e Satisfação. Esses critérios foram considerados pelos autores por acreditarem que trata-se da melhor definição que combina o processo de gerenciamento de projetos e os resultados finais de um projeto. Por Desafio, o Grupo classificou o projeto que é finalizado e operacional, mas acima do orçamento e oferece menor quantidade de recursos e funções originalmente especificadas para o produto. Por Fracasso, considera-se os projetos que não atingiram em um dos critérios estabelecidos (STANDISH GROUP, 2015).

A taxa de falha também é menor no agilismo - vide coluna Fracasso, onde o percentual global considerando todos os projetos foi de 9%, enquanto ocorreu 29% de falha nos projetos de gestão tipo Cascata.

Figura 3 – Resolução CHAOS: Métodos Ágeis *versus* Cascata

TAMANHO	MÉTODO	SUCESSO	DESAFIO	FRACASSO
De todos os tamanhos	Ágil	39%	52%	9%
	Cascata	11%	60%	29%
Projetos Grandes	Ágil	18%	59%	23%
	Cascata	3%	55%	42%
Projetos Médios	Ágil	27%	62%	25%
	Cascata	7%	68%	4%
Projetos Pequenos	Ágil	58%	38%	4%
	Cascata	44%	45%	11%

Fonte: Adaptado de STANDISH GROUP, 2015

[1]A resolução de todos os projetos de *software* do ano fiscal 2011–2015 dentro do novo banco de dados CHAOS, segmentado pelo processo ágil e método cascata.Total de projetos de *software* analisados foi superior a 10.000.

Ao estabelecer uma análise comparativa entre métodos de gerenciamento tradicionais e ágeis, o relatório CHAOS traz uma importante contribuição para todo o cenário de administração e gerenciamento de projetos e corporações, visto que a mudança nas dinâmicas de mercado provindas da aplicação difundida de tecnologias que podem otimizar entregas e lucratividade, associada ao perfil do consumidor que visa por soluções cada vez mais rápidas, elucidam as necessidades de adaptação do mercado para as novas tendências para o mercado, particularmente o de desenvolvimento de *softwares*.

Para mais detalhes sobre o método aplicado, indica-se a consulta ao relatório, que é complexo e demasiado extenso para ser elencado aqui por completo, visto que o presente trabalho o menciona apenas à titulo de curiosidade por ter sido um dos motivadores.

2.1 MÉTODOS TRADICIONAIS

Em torno de 1970 que Winston W. Royce apresentou o primeiro modelo de gestão de projetos de sistemas, conhecido como Ciclo de Vida Clássico ou Modelo em Cascata. O fundamento tradicional consiste em tornar as etapas de criação previsíveis e realizadas a partir de um plano estruturado e com detalhes. A divisão do trabalho em etapas tornou-se uma forte

característica dos métodos tradicionais de desenvolvimento de software e a confecção de documentos para alimentação de processos subsequentes, onde os times produzem itens segmentados que servirão de base para as fases subsequentes de trabalho. Para cada fase de produção, existem itens que são desenvolvidos, que podem ser desde diagramas até planos de trabalho, relatórios e testes (BASSI FILHO, 2008).

O tradicional é orientado para processos, com estilo de gerenciamento que visa controlar todas as etapas. Portanto são segmentados e dotados de muitas documentações - que são itens burocráticos - para descrever detalhes das etapas de trabalho e registrar definições relevantes ao produto. Em contra partida, ter um banco documental robusto não substitui a comunicação verbal entre a equipe, de forma que as iterações humanas são um ponto chave no processo, e por isso precisam continuar ocorrendo, mesmo que sejam produzidos muitos documentos. A comunicação é formal e a atribuição de papéis tem maior valorização pelo individualismo, onde perfis de especialistas recebem destaque. O modelo de desenvolvimento é guiado para tarefas, com etapas estabelecidas a serem seguidas (PRIKLADNICKI; WILLI; MILANI, 2014). Diversas metodologias foram elaboradas nos últimos anos, conforme ilustra Figura 2. Cabe ao leitor aprofundar os estudos nas referências bibliográficas citadas caso haja interesse em algum dos métodos que não será abordado.

O presente trabalho introduz o conteúdo sobre às metodologias tradicionais devido sua relevância para o criar produtos e também por serem base de meios de gestão aplicados na indústria de tecnologia. Serão abordados os métodos Cascata, Prototipação e *Rapid Application Development* (RAD).

2.1.1 Cascata

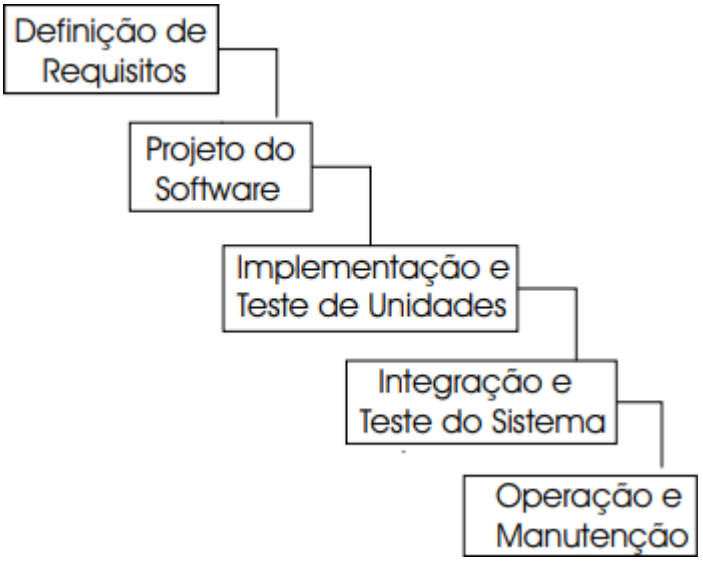
Proposto em 1970 por Royce, o Método em Cascata é uma estrutura para sequenciar atividades relacionadas à elaboração de *softwares* baseados em uma retroalimentação, onde as etapas futuras são interdependentes das etapas anteriores (ROYCE, 1987).

A Figura 4 traz os requisitos para o desenvolvimento em cascata - chamada de Método Clássico. Para Pressman (2005), o processo é linear e cada uma das atividades pré-definidassão executadas. As etapas são brevemente elucidadas a seguir.

- **Engenharia do sistema**: é a fase de engenharia básica, onde o objetivo é entender as ferramentas necessárias e indispensáveis para atingir os requisitos solicitados pelo cliente. Aqui, definem-se as tecnologias essenciais e possíveis desafios que podem ser limitantes para o projeto. Essa etapa é essencial quando o programa realizará interface com elementos diversos - como *hardwares*, bancos de dados ou usuários.

- **Avaliação dos Requisitos**: é a fase de detalhamento do produto, por isso ocorre a identificação de todas as funcionalidades do sistema. São entendidas as interfaces desejadas pelo cliente, como deve ocorrer a navegação do usuário e como o *software* deve responder mediante a iteração com ele. Toda a documentação com os detalhamentos é elaborada e enviada para análise do cliente, acontecem revisões caso seja solicitado por ele.

Figura 4 – Modelo em Cascata, ou Modelo Clássico de Desenvolvimento de *Softwares*.



Fonte: (SANTOS SOARES, 2004)

- **Projeto**: é onde ocorre a proposição de soluções. Após a fase de detalhamento, os requisitos do sistema e também as limitações estão bem desenhados e são conhecidos, portanto nessa etapa são definidas as características técnicas para construção do programa - como tipo de arquitetura, estrutura de dados, modelagens adequadas e outros. As soluções propostas devem ser totalmente personalizadas para as necessidades.

- **Programação**: é a fase onde o time de desenvolvedores atua na elaboração do código de programação que será a base para o produto desenvolvido. Toda essa etapa será feita com base nas especificações elaboradas anteriormente.

- **Testes**: são feitos testes individuais em cada elemento e também em conjunto com todo o *software*. De modo geral, são feitos testes dos aspectos internos do sistema, para garantir que as lógicas internas atendem às instruções base, e também testes de funcionalidades externas para identificar possíveis erros na aplicação final do programa e garantir que os resultados esperados serão alcançados.

- **Manutenção**: ocorre após a implementação do sistema, com objetivo de traçar futuras melhorias, como a inclusão de novas funcionalidades, correção de erros inesperados, alterações para atender ao cliente entre outros.

O problema por detrás desse modelo é sua inflexibilidade. A divisão do projeto em fases dificulta possíveis alterações - muito comuns na criação de produtos digitais. Trata-se de um modelo congelado devido o inter-travamento existente entre as etapas, por isso respostas a imprevistos podem ser demorada e até mesmo desastrosas. É um modelo aplicável em casos

onde toda a definição do produto está clara, completamente desenhada e compreendida pelo time (BASIL; TURNER, 1975).

O modelo clássico em cascata serviu de base para outros métodos de desenvolvimento - como o chamado Modelo em V (PRESSMAN; MAXIM, 2021), onde o projeto tem os passos claramente detalhados e definidos antes de o projeto ser implementado. Entretanto, em uma tentativa de eliminar os problemas relacionados ao cascata, surgiram metodologias com abordagens iterativas. Neste contexto, as abordagens de criação de *softwares* que englobam a prototipagem ganharam visibilidade à medida que se mostraram capazes de responder com mais fluidez e dinâmica aos erros ou mudanças solicitadas pelo cliente, reduzindo a quantidade de retrabalho e os riscos associados aos erros de desenho dos requisitos (FLOYD, 1984).

2.1.2 Prototipação

Uma etapa importante é a de engenharia do sistema e avaliação dos requisitos, pois são os momentos onde deve ocorrer a descrição completa do que se espera da plataforma - portanto, análises errôneas ou incompletas podem levar o projeto para um caminho não desejado, e afastar o produto do que o cliente deseja. Tal situação aumenta radicalmente o custo para correções de erros (SHULL et al., 2002).

A prototipagem é uma prática comumente utilizada para *hardwares*, componentes físicos, que serve de suporte para entender as definições de configuração necessárias ao produto em desenvolvimento. Uma vez concluído o projeto, pode haver o descarte dos itens não utilizados e apenas os módulos de sucesso são incorporados na versão final. Logo, protótipos são comumente aplicados em diversas áreas, como engenharias para o *design* de produtos para ter uma visibilidade das possíveis dificuldades que podem surgir no processo de fabricação antes de produzir o item em larga escala.

Inspirada neste conceito, a indústria de *softwares* adotou a técnica de construir modelos, simulações ou construções parciais do sistema a fim de detectar se funções estão adequadas. Neste cenário de reutilização, a aplicação da prototipação em sistemas pôde deixar o cliente mais próximo de visualizar o que poderá vir a ser a versão final do produto e trouxe facilidade nessa identificação de erros ainda durante o processo.

Para Floyd (1984), a prototipação pode ser interpretada como um processo de fato, onde cada fase é bem definida de acordo com o ciclo de vida do *software*, ou com as abordagens que influenciam toda sua construção. Assim, são quebradas etapas completas de construção do sistema em pequenas partes de complexidade menor e mais compreensível de ser trabalhada, podendo, assim, suportar o subsequenciamento de atividades usadas na construção, para atender as expectativas do cliente (KRAUSHAAR; SHIRLAND, 1985).

Portanto, a prototipação consiste em um modelo básico do sistema que é construído, testado e depois remodelado de acordo com as necessidades do cliente até que a versão final seja

atingida. O modelo é recomendado para clientes que não têm a definição detalhada dos requisitos de funcionamento do sistema, mas possuem apenas uma ideia geral do produto; também pode ser adotado quando os desenvolvedores não têm certeza sobre o algoritmo construído, do ponto de vista de eficiência e adaptabilidade.

Para Pressman (2010), o processo inicia com a coleta de informações através da Comunicação, que de modo geral é executada pelo líder do projeto em contato com o cliente para estabelecer os objetivos principais, aspectos gerais do *software* e necessidades que já estão claramente definidas. Um Plano Rápido é traçado para inicializar a modelagem com foco nos aspectos "brutos", que serão visíveis para o cliente - como interfaces, pontos do *design* e outros. A programação do sistema em si consiste na Construção do Protótipo que é seguido da Implantação, Entrega e *Feedback* do cliente. O cliente é envolvido durante todo o processo e a iteração acontece todas às vezes que erros são identificados. A Figura 5 mostra o modelo proposto.

Dois dos modelos mais difundidos no mercado são o de prototipação rápida e o piloto. No primeiro, há maior foco em explorar as possibilidades do *software* com menos rigor e burocracias formais durante o processo, portanto, trata-se de uma técnica que explora potenciais soluções para os problemas técnicos encontrados assim que aparecem. No segundo modelo, há maior ênfase em explorar o conteúdo conceitual e definições dos critérios de funcionamento sendo, ainda, incorporadas retroalimentações durante os estudos de viabilidade de implementação do programa (BOAR, 1984).

Figura 5 – Modelo de Prototipagem

Fonte: (PRESSMAN, 2010)

2.1.3 *Rapid Application Development* (RAD)

No início dos anos 90 e proposta por James Martins e Barry Boehm, o método RAD traz uma revolução no cenário de desenvolvimento de sistemas ao propor a redução do ciclo de criação para um máximo de 90 dias. A abordagem, cuja nomenclatura deriva do inglês *Rapid Application Development*, propõe a divisão do projeto em módulos para que equipes diferentes possam atuar em cada um deles ao mesmo tempo, usando o modelo em cascata com as etapas de engenharia de *software*, aplicação dos requisitos, programação, testes e manutenção (KERR; HUNTER, 1994).

Devido à estimativa de duração máxima do projeto ser dentro do intervalo de 90 dias, uma grande vantagem do RAD é a redução do tempo de projeto e redução da interdependência entre tarefas e, portanto, equipes de trabalho, devido à produção executada em módulos. Isso faz com o que o tempo de finalização do produto tenha maior previsibilidade do que em outros métodos baseados no modelo clássico - justamente pela dificuldade que o último apresenta do ponto de vista de processo.

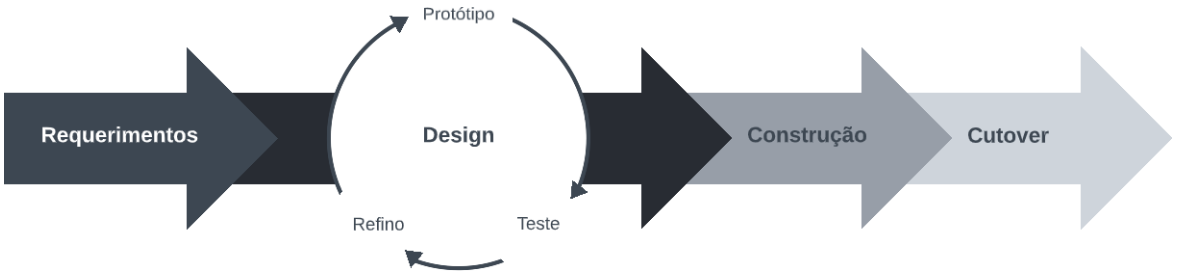
O RAD contrasta com as propostas engessadas do tradicionalismo de projetos e por isso foi inovador para a categoria e, além do paralelismo de trabalho das equipes do projeto, a execução em um menor espaço de tempo se dá graças à redução nas fases de engenharia básica, através da adoção de métodos como oficinas de trabalho ou entrevistas, e na fase de implementação final (PRESSMAN; MAXIM, 2021).

De acordo com Pressman e Maxim (2021), adotar esse método é necessário dispor de uma estrutura com ferramentas, gerenciamento e pessoas bem atualizados e desenvolvidos e, ainda, é comum que *softwares* elaborados através do RAD sejam padronizados através do uso de bibliotecas e *templates* reutilizáveis (o que traz agilidade à criação do sistema), entretanto também é comum perderem qualidade em ações que tomam mais tempo, como a análise de risco e no desempenho do sistema como um todo.

Conforme Figura 6, a metodologia inicia com a definição dos Requisitos básicos de funcionamento do sistema, seguida de definições conceituais estabelecidas junto ao cliente. Um Protótipo é construído, seguido de sua exposição à Testes para identificação de erros e melhorias que seguem para a etapa de Refino. O processo se repete até que a versão final do sistema, que passa pela Construção, é entregue. Por fim, uma revisão das atividades é feita para que ocorra o *Cutover*, que consistem na finalização técnica por completo.

Entretanto, um ponto negativo do RAD é a necessidade de mais recursos - humano e ferramental - para atuar ao mesmo tempo nas frentes de desenvolvimento. Apesar de acelerar as frentes de trabalho, o método ainda é engessado para adaptações não previstas e, por isso, não é recomendada quando os riscos técnicos são altos (BASSI FILHO, 2008). Outros pontos negativos são a interdependência entre times, necessidade de ter uma equipe com fortes habilidades técnicas entre outros. Por tudo isso, esse método é recomendado para projetos de distribuição pequena,

Figura 6 – Modelo básico - RAD



Fonte: De elaboração do autor

com poucas pessoas atuando juntas.

O RAD se consagrou por apresentar menor carga horária de dedicação quando comparado à outras metodologias clássicas, mas há especificidades de sua aplicação em pequenas equipes e surgem outros limitantes que impulsionaram o mercado a procurar por continuar a revolucionar as práticas de gestão.

2.1.4 Lean

A produção enxuta, conhecida por *Lean Manufacturing* ou somente Lean , surge na década de 40 como uma forma de readequação da indústria automotiva japonesa para melhorar a produtividade e qualidade geral dos produtos. Devido aos desafios do mercado automotivo japonês dessa época, a empresa Toyota Production Systems (TPS) surge com a proposta de uma nova forma de processos de manufatura, logística e produção com foco maior na redução máxima de desperdícios. Taiichi Ohno é considerado por muitos o pai do Sistema Toyota de Produção e, com o método, tinha intenções de agilizar a entrega do produto ao cliente, portanto todas as etapas que atrasavam a produção deveriam ser mitigados por serem considerados como desperdícios (POPPENDIECK; POPPENDIECK, 2003).

No Ocidente, James Womack e Daniel Jones foram os responsáveis pela criação do conceito Lean depois de analisarem as indústrias automobilísticas americana e japonesa (WOMACK; JONES; ROOS, 1990). Para ambos, o referido processo consiste em fazer mais com menos - menos desperdícios com transportes, movimentos, pessoas, equipamentos, espaço etc - enquanto promove cada vez mais qualidade aos clientes da maneira como desejam (WOMACK; JONES, 1997).

De acordo com Ohno (1988), existem oito tipos de desperdícios, que são conhecidos pelo acrônimo TIMWOODS:

- **Transporte (*Transportation*)**: Movimentos de pessoas, arquivos, materiais ou produtos entre atividades ou localidades que vão de encontro aos desejos do cliente;

- **Inventário (*Inventory*)**: Elaboração de documentos adicionais para processos e que são dispensáveis;
- **Movimento (*Motion*)**: Literalmente, movimentos (de cargas, pessoas, produtos, itens) que acontecem nas linhas de produção e que são aplicados em atividades que não adicionam valor para o cliente;
- **Espera (*Waiting*)**: Tempo desperdiçado em ações que param, não evoluem por esperar recursos, materiais, partes, pessoas ou informações;
- S**uperprodução (*Over-production* ou *Extra-production*)**: Produzir mais do que o necessário em curto espaço de tempo é considerado um desperdício;
- **Superprocessamento (*Over-processing*)**: Processamentos inadequados, excessivos e duplicidade são considerados trabalhos desnecessários e, portanto, não agregam valor;
- **Defeitos (*Defects*)**: Trabalhos ineficazes, itens defeituosos, ou qualquer ação indesejada durante o processo são considerados defeitos;
- **Habilidades (*Skills*)**: Habilidades subutilizadas dos envolvidos no projeto, como pessoas com alta qualificação e que atuam em atividades que extraem o máximo de seu potencial.

Portanto, os TIMWOODS - também conhecido por *DowTime* - recebem atenção dentro do Lean para que sejam eliminados a fim de aumentar a eficiência, fluxo de trabalho e velocidade de entrega final do produto.

A chamada Cadeia de Valores no Lean apresenta tudo que o projeto exige para que o produto seja executado conforme desejo do cliente e vão desde a matéria prima até as definições técnicas indispensáveis para criar o produto da maneira ideal para o cliente, evitando desperdícios no caminho. Assim sendo, para Womack e Jones (2003), os valores - ou princípios de pensamento Lean - são elencados a seguir e ilustrados na Figura 7.

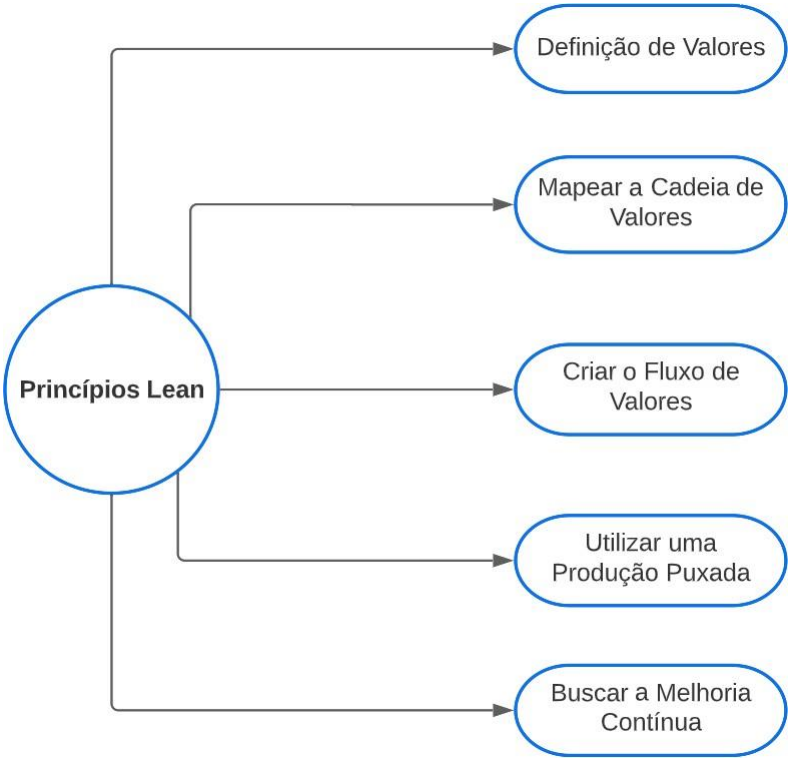
- **Definir os Valores**: Para entender como o projeto deve ser direcionado, o primeiro passo é definir o que é importante e indispensável para solução do produto - esses são os valores. Manter o foco nos valores é indispensável, uma vez que toda atividade que não os atende é dispensável no Lean;
- **Mapear a Cadeia de Valores**: O segundo princípio consiste em identificar e mapear a cadeia de valores com objetivo de direcionar as atividades para irem no sentido de atender aos desejos do cliente. Atividades que não agregam valor são consideradas desperdícios, que existem em duas categorias - sem valor agregado, porém necessário (e, por isso, deve ser reduzido ao máximo) e sem valor e desnecessário (portanto, deve ser eliminado). Dessa forma, pode-se garantir que o sistema estará o mais próximo possível do desejo do cliente.

- **Criar o Fluxo de Valores**: Após remover os erros, espera-se que o fluxo corra da maneira adequada e sem interrupções ou atrasos. Algumas técnicas para garantir isso são a reconfiguração das etapas de produção, como na repartição das mesmas; criação de departamentos multidisciplinares; envolvimento de colaboradores com diversas habilidades e alta capacidade de adaptação, investir em treinamentos profissionais se necessário, entre outras.

- **Utilizar uma Produção Puxada**: Enquanto a produção empurrada trabalha com previsões de demanda e a produção começa antes mesmo de um cliente adquirir o produto, a produção puxada garante que a criação do produto comece somente após o pedido do cliente. Esse princípio tem por objetivo limitar o trabalho aplicado em atividades que estão fora da cadeira de valor, enquanto também garante que as informações, materiais e requisitos estarão disponíveis de maneira fácil e rápida para os envolvidos.

- **Buscar Perfeição/ Melhoria Contínua**: A constante melhoria é derivada da busca pela perfeição nos processos, na relação dos times, na qualidade dos produtos e serviços etc. No setor de tecnologia, exemplos de métodos de melhoria contínua são os chamados Kaizen e PDCA (do inglês, *Plan*, *Do*, *Check*, *Act*). O primeiro deles, em resumo, foca na melhoria contínua de toda a corporação e envolve todos os níveis hierárquicos da liderança; o segundo, se divide em ciclos que visam aumentar o nível da gestão interna de processos e otimizar controles de gestão.

Figura 7 – Os Cinco Princípios Lean.



Fonte: De elaboração do autor.

Para aprofundar conceitos do Lean , como Cadeia de Valor, Produções Empurrada e Puxada, Kaizen, PDCA e para conhecer outros como Teoria das Restrições, Teoria das Filas, *Just in Time* e os outros diversos pilares da metodologia, etc, a obra de Ohno (1988) e as outras referências bibliográficas aqui citadas podem ser consultadas pela pessoa interessada. Por se tratar de uma filosofia muito robusta e extensa, este estudo coloca foco em apresentar a aplicação do Lean no desenvolvimento de *softwares*.

A mentalidade Lean é importante porque pode reduzir em 1 milhão por unidades a taxa de defeitos (SCHONBERGER, 2008); pode reduzir significativamente o tempo para entregar novos produtos enquanto traz redução substancial dos custos envolvidos (WOMACK; JONES, 1994) (ABEGGLEN, 1986); pode ao menos dobrar a produtividade de atividades de manufatura e serviços de operação (WOMACK; JONES; ROOS, 1990) (OHNO, 1988). As evidências são diversas, visto que indústrias como Toyota (Japão) e Porsche (Alemanha) mostram que a aplicação dos conceitos Lean podem aprimorar significativamente os níveis de uma empresa (WOMACK; JONES, 1997).

Com o passar dos anos, a aplicação de produção enxuta passou a ser aplicada em outras disciplinas e o modelo é utilizado em muitas indústrias de grande porte, inclusive no setor de sistemas digitais. É vital entender que, para os *softwares*, ao invés de produzir em larga escala toda a documentação necessária e somente depois evoluir para a criação do produto, cada etapa passa a ser finalizada e deve ser aprovada pelos desenvolvedores ou pessoas envolvidas no projeto. Caso erros ou inconsistências sejam encontrados, a equipe técnica deve fazer uma nova análise e aplicar uma correção. Isso garante que os desenvolvedores terão tempo para checar os erros logo após finalizarem seu trabalho e é possível realizar análises constantes do desempenho para trazer os pontos de destaque nos problemas organizacionais, como impactos na motivação, retenção e treinamento do time.

A aplicação do Lean em projetos de criação de sistemas digitais é mostrada em diversos trabalhos, como de Fagan (1979) que apresenta uma abordagem refinada e mostra a aplicação dos métodos de controle de processos e estatística aplicada ao controle de qualidade, e Gilb (1988) que mostra o desenvolvimento evolutivo de sistemas com abordagens Lean .

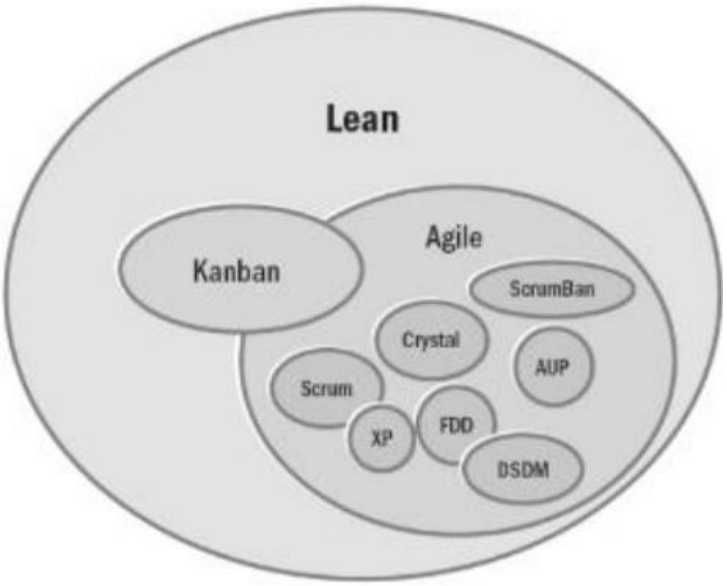
A mudança de paradigmas para entender a aplicação do Lean aos projetos digitais é entender que as especificações de um sistema, quando ainda estão em um papel, têm o mesmo efeito de um estoque em uma linha de produção. Nos métodos tradicionais o desperdício real ocorre, não somente quando há desperdício de papel para gerar documentações preliminares, mas quando os erros que podem existir na prática ficam escondidos dentro das documentações, justamente pela ausência da criação. Enquanto ainda no papel, detalhes preciosos podem ser esquecidos, ou, ainda, podem simplesmente mudar ao longo do caminho - e quando a documentação estiver pronta, já não estará mais condizente com a realidade atual. Não há uma retroalimentação rápida e constante para correção rápida dos erros.

Considerados como os precursores da aplicação dos conceitos Lean à produção de

softwares, Mary e Tom Poppendick apresentam um conjunto de propostas para essa aplicação, incluindo ferramentas para identificação de erros e criação de soluções. Middleton e Sutton (2005) também analisam a aplicação do Lean ao setor de tecnologia, com foco na combinação junto de outras estratégias - como a reutilização do *softwares* e a metodologia *eXtreme Programming* (XP). Dentre as diversas ferramentas, destaca-se o Kanban que tem sido aplicado em diversos projetos de *softwares*.

Por fim, com a publicação do Manifesto Ágil em 2001, que chamou atenção de diversos mercados para as possibilidades e ganhos trazidos pelas formas agilistas de gestão, que remodelaram a forma tradicional - embasada no modelo Cascata e suas variantes - e também no Lean. Vale destacar que toda a atualização das técnicas de gestão foram importantes para consagrar as metodologias ágeis de hoje. Por isso, o presente trabalho apresenta uma visão macro da filosofia Lean (supracitada em 2.1.4), do método Scrum e da ferramenta Kanban - que consistem em metodologias bastante difundidas em projetos de criação de sistemas digitais e que surgiu como suporte à gestão Lean. A Figura 8 mostra a relação entre Lean, métodos ágeis (*Agile*) e o Kanban.

Figura 8 – Relação Lean, Métodos Ágeis e Kanban



Fonte: (ALLIANCE, 2017)

2.2 MÉTODOS ÁGEIS

Em meados dos anos 90, as metodologias ágeis surgem como alternativas às metodologias tradicionais de desenvolvimento de produtos digitais. Como o próprio nome sugere, há um aumento da rapidez de entrega do produto através da desburocratização dos processos - onde há menor concentração de esforço na elaboração de documentos - e maior concentração de trabalho no projeto do *software*. Portanto, as metodologias surgem como uma resposta para as rápidas mudanças nas demandas dos produtos (RISING LINDA E JANOFF, 2000).

Influenciadas pelos modelos de estratégia gerenciais desenvolvidos pelos japoneses Senge (1990), Takeuchi e Nonaka (2004) que se inspiraram nas melhores práticas de gerenciamento industriais - principalmente em *Lean Manufacturing*, que foram utilizadas por Honda e Toyota - surgem as técnicas de gestão ágil.

Muito popular nos dias atuais, as metodologias ágeis têm crescido no mercado de trabalho e não é incomum serem encontradas em ambientes de projetos que não os de *softwares* digitais. Entretanto, ainda sim, são altamente difundidas no ramo da tecnologia e têm ganhado espaços cada vez maiores.

A fundamentação do agilismo consiste em entregar *softwares* adaptativos e de alta qualidade que podem ser criados por equipes multidisiciplinares apenas aplicando conceitos de melhoria contínua ao projeto, associados à testes constantes para que a identificação e correção de erros sejam feitas do modo mais rápido possível. São orientados para pessoas, diferentemente das formas tradicionais, portanto a elaboração de documentos não essenciais ao cliente ou criação do produto são dispensáveis. Por sua orientação, é natural do agilismo ter estilo de gerenciamento horizontal, focado em colaboração. A comunicação é fluida e pode ocorrer de formas diversas, não havendo barreiras formais e podendo ser definidas por cada equipe de trabalho. Times autogerenciáveis são o tipo de recurso humano valorizado, que também conta com uma gestão do conhecimento menos engessada e ciclos de projeto voltados para funcionalidades essenciais do produto - definidas com base nos valores do cliente. Modos iterativos e incrementais de entrega são, portanto, a forma de criação dentro da estrutura organizacional orgânica, com alta flexibilidade e incentivo à cooperação social existente nas Ágeis (PRIKLADNICKI; WILLI; MILANI, 2014).

2.2.1 O Manifesto Ágil

Elaborado por um time de dezessete desenvolvedores de *software* e se apresentando como uma metodologia alternativa para os processos tradicionais de desenvolvimento, o Manifesto Ágil (em inglês, *Agile Manifesto*) surge no ano de 2001 com abordagens novas para o processo de criação de sistemas, com foco em tornar possível a entrega mais rápida do produto e também com maior adaptação aos erros ou problemas encontrados. A proposta do Manifesto causou impacto no mercado de desenvolvimento de *softwares* ao tratar as fases do projeto com grande desburocratização e com foco obsessivos nas entregas mais primordiais para atender as necessidades do cliente.

São quatro os valores do Manifesto Ágil, que visam valorizar:

- Indivíduos e interações mais que processos e ferramentas;
- Sistema em funcionamento mais do que documentação pronta;
- Colaboração com o cliente mais do que negociações contratuais;

- Respostas às mudanças mais do que seguir um plano estático.

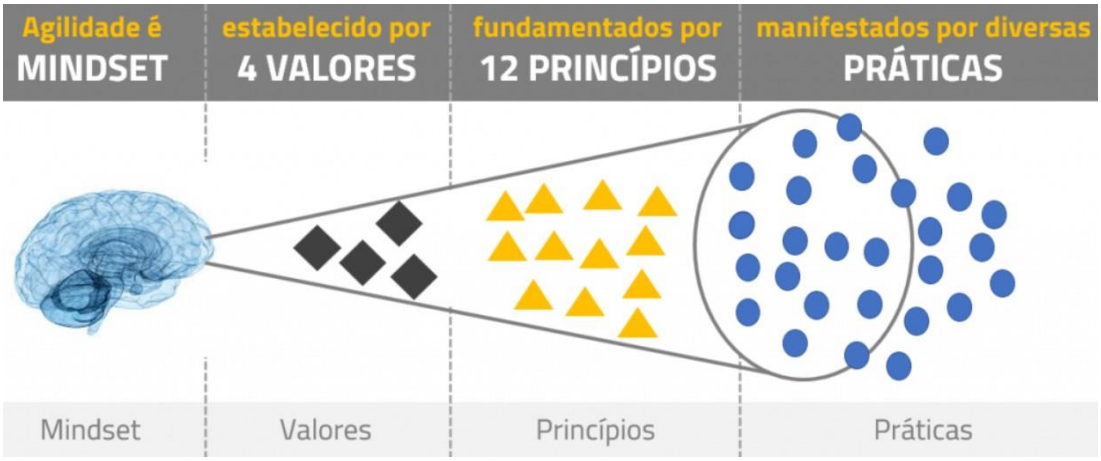
Isso não significa que os Métodos Ágeis não valorizam os processos, ferramentas, documentações, negociações e contratos; apenas ressalta que o foco dos times ágeis são nas primeiras variáveis citadas - indivíduos, interações, sistema funcional, colaboração e dinamismo através da rápida resposta às mudanças.

Sato (2007) realiza uma análise sobre os doze princípios do Manifesto que suportam a organização da metodologia - que seguem abaixo de forma resumida. A relação entre valores, princípios e práticas ágeis pode ser verificada na Figura 9.

- Satisfação do cliente através de entregas ágeis e contínuas é a prioridade;
- Alterações nos requisitos do programa são aceitas, ainda que em estágios avançados de desenvolvimento;
- *Software* que funciona é entregue com frequência, em períodos que podem variar para atender ao tempo estabelecido pelos envolvidos;

  As pessoas farão o trabalho se o projeto for formado por pessoas motivadas, em um ambiente com suporte e confiança;
- As pessoas relacionadas ao projeto devem trabalhar juntas, independentemente do local;
- A comunicação horizontal entre pessoas é a forma mais eficiente de trocar informações entre um time;
- O funcionamento do *software* é a principal métrica do progresso;
- Métodos ágeis têm objetivo de promover um desenvolvimento sustentável e as equipes devem ser capazes de trabalhar constantemente, por tempo indeterminado, em ritmo constante;
- É possível aprimorar a agilidade do projeto através de atenção constante à excelência técnica e *design* do sistema;
- Simplicidade é a arte de maximizar a redução do volume de trabalho que deve-se ter e isso é essencial;
- Equipes autogerenciáveis apresentam o melhor resultado de trabalho;
- Em períodos de tempo fixados, a equipe se reúne para refletir sobre seu desempenho e como podem tornar-se mais eficiente.

Figura 9 – Relação entre valores, princípios e práticas ágeis



Fonte: (ATÉ O MOMENTO, 2019)

Por trazer grande foco nas pessoas envolvidas no processo, o Manifesto Ágil provê dinamismo ao trabalho em equipe, mantendo as pessoas motivadas e com acesso umas às outras, quebrando hierarquias que podem distanciar os indivíduos, prejudicando seu desempenho ou, ainda, causando impactos negativos no projeto.

Essa característica faz com que o agilismo seja interessante método de trabalho não somente para aplicação em *softwares*, mas também para diversos outros tipos de projetos e processos que envolvam pessoas em trabalhos em grupo, mediante a adaptação para a realidade do referido trabalho.

2.2.2 Scrum

Desenvolvido em 1993 com base nas teorias de Takeuchi e Nonanaka (1996) que discutem, entre outros pontos, as vantagens de pequenos times de desenvolvimento (TAKEUCHI H. E NONAKA, 1986), o Scrum tem grande foco no cliente, sendo bastante popular nos dias atuais, também é de fácil aplicação a pequenas equipes de trabalho - por isso foi alvo do presente estudo. De acordo com o estudo *15th State of Agile Report*, publicado em 2021, mostra que 66% das equipes de *softwares* utilizam o Scrum.

O nome Scrum foi inspirado em um movimento do jogo americano chamado *rugby*, onde os jogadores se reúnem brevemente para iniciar uma jogada com objetivo de recolocar a bola em jogo - este nome foi definido pela primeira vez em 1986 por Takeuchi e Nonaka. A equipe com objetivo final é entregar o sistema de acordo com as necessidades do cliente e com qualidade.

Os modelos tradicionais são engessados à mudanças por adotarem um processo definido, mas estudiosos da área de gerenciamento, como Ken Schwaber, Jeff Sutherland e Mike Beedle - que lapidaram o Scrum para o mundo ocidental, notaram que um processo de produção capaz

de absorver as variações do processo e que fomentasse à liberdade de adequação dos meios de resolução para personalizar as soluções dos problemas, poderia ser uma abordagem mais adequada para os projetos de *softwares*. Assim sendo, o Scrum utiliza-se do auto-gerenciamento dos times para o sucesso de um projeto.

Deve-se destacar o cenário de aplicação recomendado para o Scrum, que consiste em um *framework* (do inglês, metodologia) para projetos complexos, é recomentado para negócios com equipes que são estáveis e altamente adaptáveis, de forma que não deve haver a alteração durante a evolução do projeto para que não ocorra perda de capital intelectual, visto a não priorização da elaboração de procedimentos e documentos no agilismo.

Para entender o processo de desenvolvimento é necessário conhecer, de forma geral, as fases e os papéis e responsabilidades envolvidos no Scrum, que tem objetivo de definir um processo de projeto com foco nas pessoas (SCHWABER; BEEDLE, 2002). Adaptado em 1993 no mundo ocidental por Jeff Sutherland, Ken Schwaber e Mike Beedle, tem como base as características a seguir (SCHWABER, 1997):

- Pequenos times;
- Prazos flexíveis;
- Revisões frequentes;
- Cooperação;
- Resultados flexíveis;
- Orientação à objetos.

De acordo com Mundim et al. (2002), para desenvolver um produto é necessário ter um time com certos tipos de informação e habilidades, pois o processo está relacionado à todos os papéis de uma companhia, portanto trata-se de uma atividade multidisciplinar e cada projeto tem suas particularidades e conhecimento prévio únicos e indispensáveis.

Assim sendo, o método não exige técnicas específicas para as fases de desenvolvimento, há apenas regras de trabalho para os grupos e práticas de gerenciamento que devem ser adotadas para o sucesso do projeto. Abaixo elas serão introduzidas separadas em três categorias, resumidas conforme Figura 10.

1. Artefatos

   Os artefatos representam o trabalho elaborado para trazer transparência ao processo e manter todos do time com o mesmo entendimento. Existem três artefatos, elencados a seguir.

- ***Product Backlog***: É a lista de atividades que devem ser executadas durante a criação do produto. Essa prática é o passo inicial e deve haver um levantamento de todos os itens necessários. É realizado através de reuniões com o time, incluindo os clientes, sócios e parceiros, para definir as necessidades do negócio e as técnicas que, em um primeiro momento, serão necessárias para atingir os objetivos do sistema.
- ***Sprint Backlog***: É o conjunto de ações do *Backlog* que serão realizados. Cada *Sprint* corresponde a uma sessão de trabalho e abarca as fases de execução, onde a lista de priorizações (*backlog*) será trabalhada. Essa fase pode ser encontrada na literatura com a nomenclatura de"caixa preta", pois eventos não previsíveis podem acontecer. Pode durar de uma a quatro semanas. De acordo com Abrahamsson (2002), cada *sprint* inclui as fases de criação tradicional de *software* onde ocorrem as definições dos requerimentos, detalhamentos, *design*, desenvolvimento e entrega. Pode, ainda, haver diversas equipes de trabalho atuando ao mesmo tempo. As durações devem ser coerentes, respeitando os prazos, e uma nova etapa se inicia imediatamente após a finalização da anterior. É possível ter previsão do projeto através do acompanhamento das *Sprints* e realizar adaptações necessárias à evolução em direção à meta através da etapa de Revisão da *Sprint*.
- **Gráficos *Burndown***: Permite acompanhar a evolução do trabalho durante o tempo disponível para executá-lo. É uma ferramenta visual que pode englobar diversos pontos do desenvolvimento, com a prototipação de um sistema. Ferramentas como os *softwares* Jira e Azure podem suportar a elaboração dos gráficos.

2. Cerimônia, ou Reuniões

- **Planejamento da *Sprint***: É a reunião para criar o plano de trabalho colaborativo para toda a equipe de desenvolvimento. Essa atividade tem duração definida de acordo com o tempo de duração da sessão de trabalho, mas pode chegar a um máximo de oito horas para planejamento de uma *Sprint* de quatro semanas. Questões primordiais que devem ser respondidas nessa etapa é o que pode ser entregue e qual o trabalho exigido para a entrega na próxima *sprint* (SCHWABER; SUTHERLAND, 2013).
- **Revisão da *Sprint***: É executada sempre no final de cada *Sprint* para medir as evoluções e traçar as melhorias e otimizações que permitirão a melhoria contínua do processo. A depender das conclusões da equipe de trabalho, pode haver alteração no *Backlog* do produto, com foco em alcançar maior qualidade e valor. É uma atividade informal visando a colaboração entre o time. A duração limite para a Revisão é de quatro horas para uma *Sprint* de quatro semanas.
- ***Daily***: Em português, significa Reunião Diária e consiste na atividade que reúne todo o time para definição das tarefas diárias, ou para comunicação dos resultados do dia anterior. Também pode ser chamada de "Reunião em Pé"(do inglês, *Stand*

Up Meeting), pois é feita de maneira rápida. A *Daily* deve ter obrigatoriamente 15 minutos no Scrum e é apenas para sincronização das atividades do plano de trabalho. Perguntas primordiais sobre o trabalho feito no dia anterior e sua validade para o dia atual são realizadas, juntas de análises sobre os obstáculos que podem impedir a atuação do time (SCHWABER; SUTHERLAND, 2013). Portanto, a ação serve para sintonizar, inspecionar e acompanhar a evolução da *Sprint*.

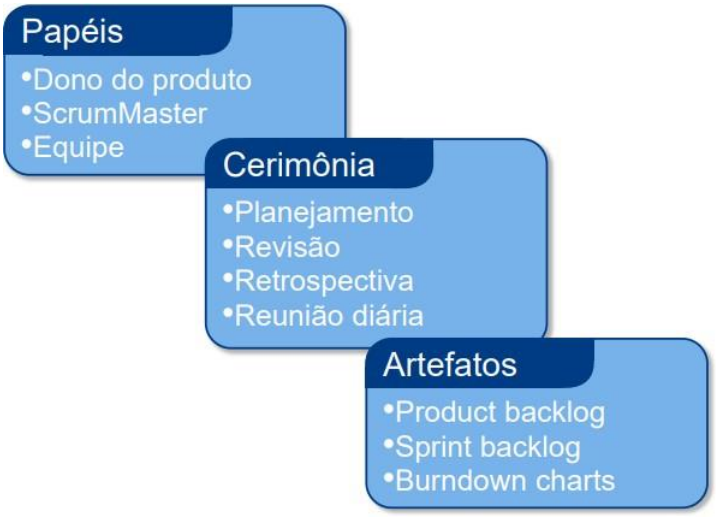
- **Retrospectiva da *Sprint***: Tem objetivo de fazer uma revisão do trabalho do time e criar planos de melhorias. Ocorre depois da Revisão e com duração máxima de três horas para uma *Sprint* de quatro semanas. O foco da Retrospectiva é analisar quesitos humanos do projeto, como comportamento das pessoas, relacionamentos, resposta ao uso de ferramentas e processos, a fim de elencar as melhorias também à equipe de trabalho e não somente ao *software* em desenvolvimento.

3. Papéis

Considerando os papéis e responsabilidades, a metodologia traz os seguintes personagens:

- ***Product Owner* (PO) ou Dono do Produto**: Trata-se do representante do cliente que acompanhará as etapas do projeto para garantir que o projeto está direcionado conforme as necessidades do cliente. Deve trazer a visão do produto e expectativas para o sistema, exercendo o gerenciamento do *backlog* de longo prazo e auxiliar na priorização das entregas de curto prazo. Como define Abrahamsson (2002), é oficialmente o responsável por gerenciar, controlar e planejar as atividades de definição do sistema e é quem define as tarefas finais que serão executadas nas próximas *sprints*. Portanto, é responsável por gerenciar e ordenar os itens do *Backlog*, bem como garantir que a equipe de desenvolvimento tenha acesso às informações e atue adequadamente. O PO é uma pessoa, e não um comitê (SCHWABER; SUTHERLAND, 2013).
- **Scrum *Master***: É responsável por garantir que a metodologia está corretamente aplicada e é entendida por todos os envolvidos. Atua como um consultor a fim de garantir que o projeto está caminhando conforme as práticas, valores e pilares necessários do método. Mantém interações com todos os outros papéis e deve possuir conhecimentos sobre o Scrum, bem como experiências de trabalho. É responsável por assegurar que imprevistos ou erros sejam sanados e que o time se mantenha engajado, garantindo um alto desempenho das equipes.
- **Time Scrum**: É o time de desenvolvedores que irá suportar toda a parte técnica necessária à criação do *software*, com autoridade para se auto-gerenciar e organizar para atingir as metas de cada sessão de trabalho. É envolvido nas etapas de definição do *backlog*.

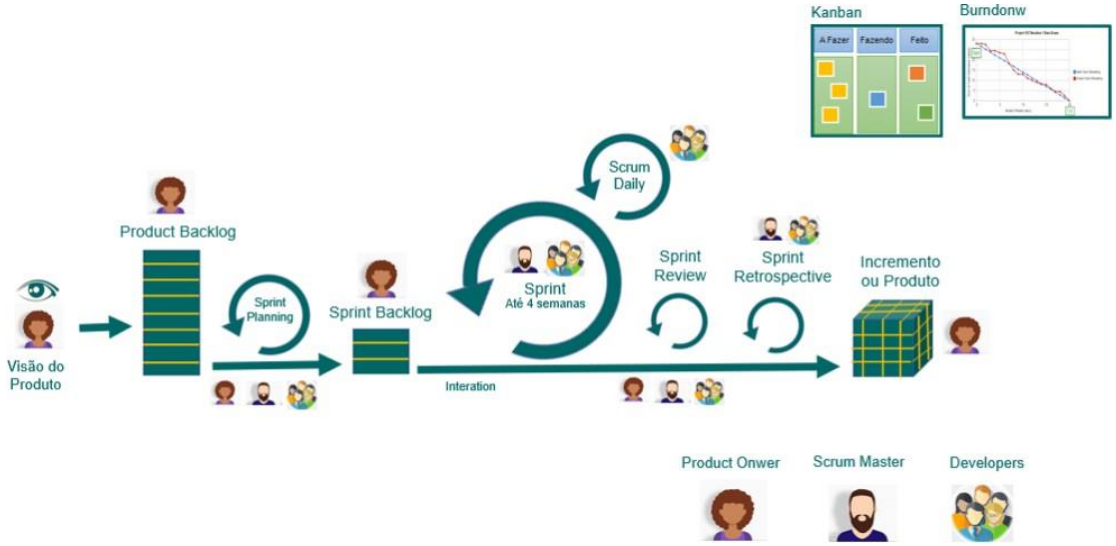
Figura 10 – Papéis, artefatos e cerimônias no Scrum



Fonte: (COHN, 2018)

A Figura 11 ilustra a dinâmica geral do método Scrum.

Figura 11 – Modelo geral da dinâmica da metodologia Scrum



Fonte: (ANALISTA EXPERT, 2020)

No agilismo é comum encontrar variações dos métodos de gestão, onde uma técnica é mesclada com outra para trazer melhores resultados. Um bom exemplo é a aplicação da ferramenta desenvolvida pelo Lean e que recebe o nome de Kanban, que pode ser inclusive utilizada junto à gestão Scrum. Quando isso acontece, o método recebe o nome popular de Scrumban. Como é possível mesclar o Kanban com outras técnicas, o que a faz ser versátil, o

presente trabalho traz maiores informações sobre a ferramenta.

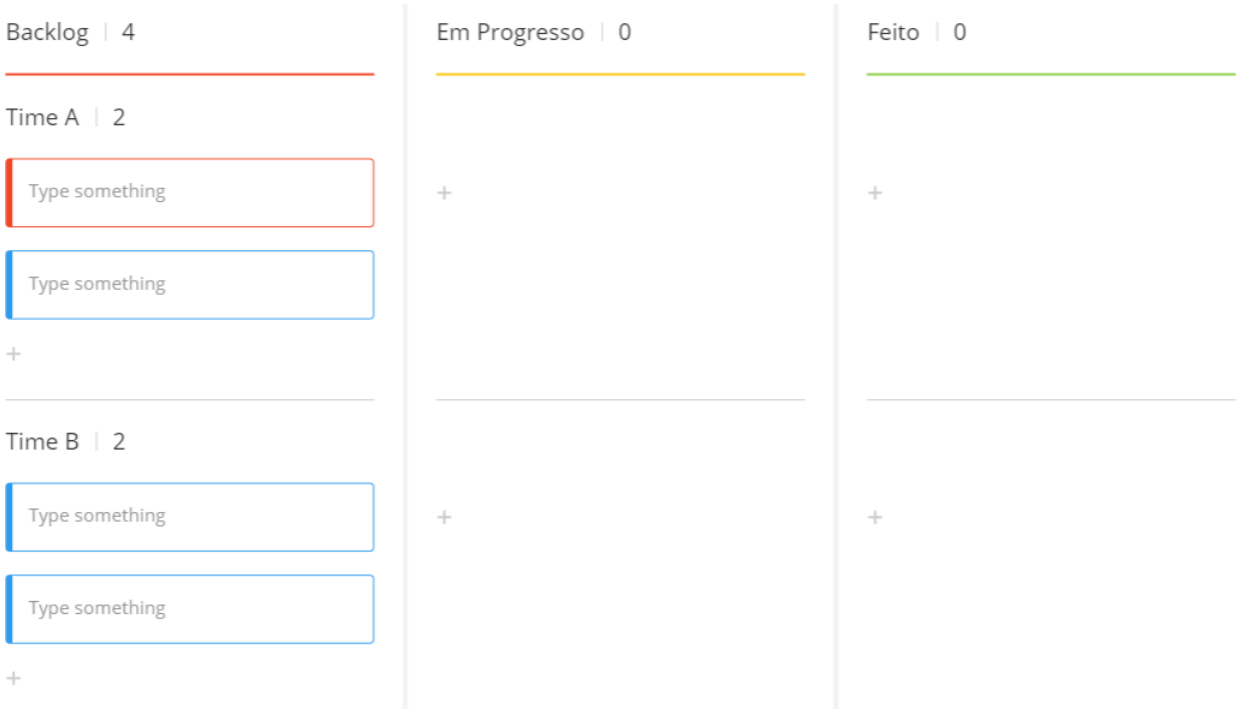
2.2.3 Kanban

O Kanban é uma das ferramentas para gerenciamento de operações de produção (LIKER, 2004) e é uma abordagem recente das disciplinas de gerenciamento ágil e Lean para desenvolvimento de *softwares*. Trata-se de um meio para melhorar a criação de sistemas digitais e foi desenvolvido pela Toyota com objetivo de auxiliar o controle de inventários, da produção e de componentes da cadeia de produção - como no recebimento de matérias primas. A metodologia surge como um mecanismo de controle de fluxos para gerenciar o tempo e volumes de produção - um suporte à técnica *just in time* (GRAVES; KONOPKA; MILNE, 1995). Kanban deriva do idioma japonês e significa "quadro de sinais", um conceito relacionado à produção Lean . De acordo com Taiichi Ohno, mesmo criador do Lean , a utilização do Kanban não se limita ao controle de inventários, mas pode ser considerado como uma forma de visualização do trabalho, facilitando a ação das equipes e, consequentemente, otimizando o projeto e impactando nos resultados por trazer a redução de desperdícios (OHNO, 1988).

O Kanban se popularizou mundialmente devido a utilização de cartões para controle dos processos - por isso também é comum se referir à técnica com "cartões". É composto por uma estrutura de ferramentas que tem objetivo de facilitar a gestão de projetos, otimizando os resultados durante o processo de elaboração de *softwares*. Suas limitações são abordadas em diversas literaturas - como de Monden (1984) e Ohno (1982): situações com instabilidade de demanda, tempo de processamento e falta de métodos estabelecidos, além de exigirem longos tempos de configuração e variedades de itens. Por isso, é necessário ter um cenário com as definições especificadas.

O quadro Kanban tem objetivo de tornar visíveis as fases de execução do projeto. É comumente composto por colunas que irão segregar cada uma dessas fases. A nomenclatura aplicada para elas pode variar, mas o processo proposto exige ao menos três fases que recebem o nome do *status* de avanço, sendo elas: A Fazer (*Backlog*), Em Progresso (*In Progress*) e Feitas (*Done*). A Figura 12 elucida um modelo clássico de Kanban para controle de atividades.

Figura 12 – Quadro Kanban Clássico



Fonte: De elaboração do autor.

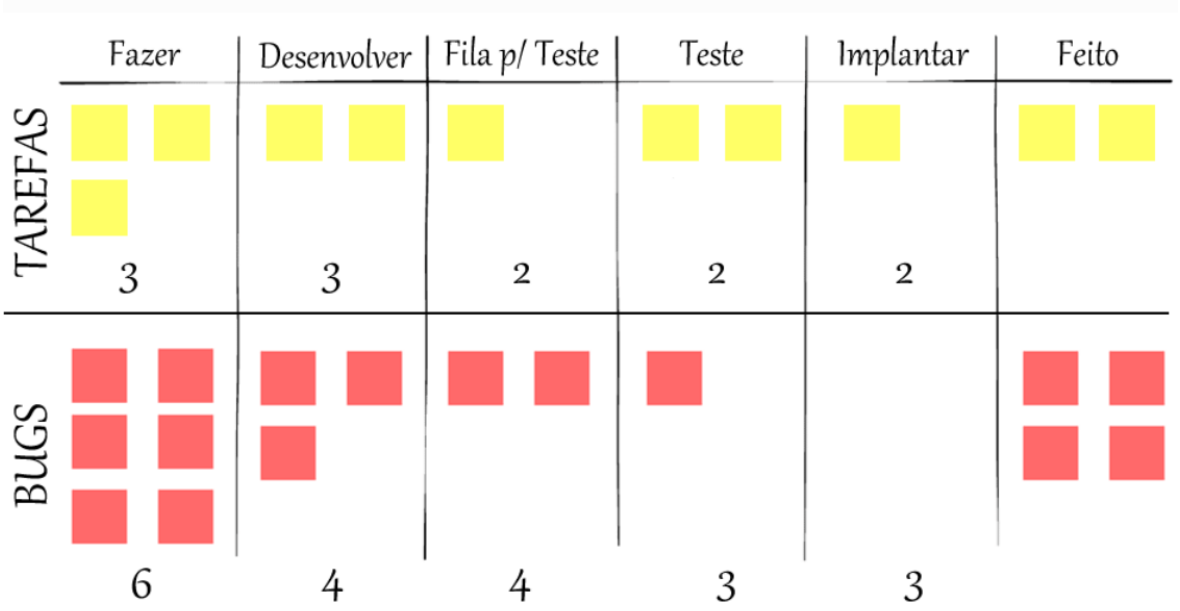
2.2.4 Lean-Kanban

O Kanban é uma ferramenta que permite a customização personalizada de monitoramento de acordo com as necessidades. Também é bastante comum contatar com outros níveis como Em Revisão, Em Teste, Em Pausa (Bloqueados) - ou variações desses nomes, conforme mostra a Figura 13.

Pelo fato de a metodologia Lean não representar um percentual muito grande de aplicações Widman, Hua e Ross em seu estudo *Lean Principles in Software Development Process - A Case of Study*, publicado em 2010, implementam o Lean como forma de testar a forma de gestão. A pesquisa foi realizada em uma empresa virtual proprietária de uma plataforma digital de relacionamentos virtuais com mais de 25 milhões de inscritos e tem 90% de sua receita vinda de vendas realizadas para usuários que adquirem itens digitais em um catálogo com mais de 1,8 milhões de itens. Em 2008, a empresa foi premiada no *Virtual Worlds Innovation Award* e também nomeada como uma empresa de destaque por um programa do Vale do Silício chamado *Silicon Valley Technology Gast 50* (WIDMAN; HUA; ROSS, 2010).

Os criadores da plataforma IMVU Inc. decidiram aplicar conceitos de Lean em seu desenvolvimento depois de já terem testado a criação de outros conteúdos digitais sem técnicas adequadas de gerenciamento e acabaram obtendo insucesso em projetos anteriores (WIDMAN; HUA; ROSS, 2010). Portanto, na IMVU Inc. os criadores decidiram empregar uma mentalidade

Figura 13 – Quadro Kanban Personalizado



Fonte: (SINDICONT, 2018)

de despachar o máximo possível de entregas e depois coletar a opinião do cliente. Em essência, lançar um subproduto que permitia a empresa entender o caminho a seguir para atender aos desejos dos usuários sem desperdiçar tempo com superproduções ou na criação de funcionalidades que não necessariamente os clientes iriam absorver e aprovar.

A aplicação do Kanban se deu em conjunto com a chamada Andon Cord, que consiste em um sistema de gerenciamento visual inspirado nas linhas de manufatura automobilística da Toyota, onde havia uma corda para ser puxada quando ajuda ou suporte eram necessários. Puxar a corda muitas vezes é considerado algo positivo, pois com as falhas que ocorrem no caminho é possível aprender novas formas de melhoria que levariam aperfeiçoamento futuro. Na criação de *softwares* é comum que o projeto siga adianta mesmo com a existência de erros, que podem ser "carregados"durante todo o processo e automação de testes pode auxiliar a agilizar a correção de falhas. Foram utilizadas ferramentas do Lean, como de Cinco Porquês, Kanban e Andon Cord - todas criadas pelo pai do toyotismo em 1988.

Vale pontuar que, utilizar o Andon Cord na etapa de testes faz com que seja enviado um sinal para os desenvolvedores ou líderes do projeto caso haja falhas no sistema, assim os responsáveis podem direcionar as tratativas adequadas mais rapidamente. Se o problema não for solucionado em um intervalo temporal específico, todo o projeto para e a equipe se reúne para direcionar o problema até que tudo esteja claro e ele seja resolvido. Isso garante que as lições serão aprendidas e os erros não serão repetidos no processo. Outra aplicação possível para o Andon Cord é nos chamados Testes A/B, que testam duas variáveis por vez a fim de compará-las e permanecer com a versão mais otimizada possível.

O uso do Kanban com o Andon Cord fez com que cada desenvolvedor atuasse somente em uma tarefa por vez e fossem proibidos de evoluir para outras frentes de trabalho até que o código de programação seja aprovado no teste automatizado. Isso significa que o time de desenvolvimento passa a ter janelas de tempo livre enquanto os testes ocorrem, ao passo que também garante que o módulo de programação estará completo e sem erros antes de que outra atividade seja executada - mantendo uma alta qualidade do produto.

Aconteceram, ainda, problemas de infraestrutura, visto que os testes automatizados consistiram em uma fase muito difícil para a equipe, pois com o crescimento constante das bases de dados, houve também um aumento da complexidade dos testes que, por sua vez, tornou evidente a necessidade de investir em infraestrutura visto que a força computacional passou a crescer em uma escala exponencial (WIDMAN; HUA; ROSS, 2010).

Em contra partida, o volume de testes que foi possível realizar em um curto espaço de tempo aumentou na escala de milhões - mais especificamente, 1:2.000.000,00. Esse resultado torna evidente a capacidade que boas metodologias aliadas a ferramentas adequadas podem trazer para um sistema, que no referido caso teve uma potencialização grande da qualidade do *software*.

3 DESENVOLVIMENTO: ESTUDO DE CASO SCRUM

A fim de trazer maior visibilidade sobre os processos de trabalho que são base das metodologias ágeis na prática, o presente estudo traz a análise de um estudo de caso para avaliar qualitativamente os processos de gestão e por isso são abordados aspectos dos métodos em si, papéis e responsabilidades dos envolvidos.

Como os métodos ágeis priorizam pessoast, os autores do estudo elaboraram pesquisas de opinião que foram aplicadas aos integrantes das equipes de desenvolvimento para conseguir medir o grau de satisfação obtido através do referido método de gestão.

Cabe ressaltar que a análise puramente quantitativa para a comparação entre métodos ágeis e tradicionais de gestão apresentaram dificuldades metodológicas. Idealmente, para normalização completa e puramente quantitativa, seria necessário acompanhar dois projetos iguais, ocorrendo simultaneamente ou não, durante todo o processo de criação de um mesmo produto, porém com aplicação de metodologias diferentes. Esse contexto de comparação de métodos no mundo corporativo é muito complexo e de difícil implementação por diversos motivos: encontrar um ambiente com recursos disponíveis para comparação de duas formas diferentes de gerencia- mento apresenta alto custo e, ainda, exigiria que dois produtos completamente idênticos fossem desenvolvidos ao mesmo tempo e isso, de modo geral, não apresentaria vantagem estratégica competitiva do ponto de visto do mercado devido à redundância de recursos - somente assim o critério de análise científica justificaria uma comparação dos métodos tradicional e ágil nos índices do projeto - como quantidade produzida, custos de produção, tempo, entregas feitas e pendentes, tempos de entrega e efetividade dos ciclos de desenvolvimento, quantidade de erros/retrabalhos e outros.

Essa dificuldade existe devido à característica diferenciada do produto digital, que não é elaborado por processos normalizados de manufatura e pode apresentar diferentes erros durante sua criação, e eles podem ser diversos pois envolvem códigos de programação com uma imensa possibilidade de modos de serem criados - e pessoas que fazem o processo criativo, por isso - ainda que aplicando a mesma sintaxe lógica ao algoritmo - há inúmeras formas de construção de um mesmo *software*.

Não obstante, é relevante a observação da aplicação das formas de trabalho para entender o contexto em que apresentam maior sucesso e como as metodologias podem impactar as pessoas de um projeto.

Carvalho e Mello (2012) realizaram um estudo com estratégia de pesquisa-ação de natureza aplicada em uma empresa de pequeno porte responsável pela criação de sistemas digitais, conforme mostra o trabalho "Aplicação do método ágil Scrum no desenvolvimento de produtos de *software* em uma pequena empresa de base tecnológica". O interesse dos autores foi em observar a prática de aplicação do Scrum e os resultados na resolução de obstáculos reais.

Os objetivos da pesquisa descritiva foi "descrever as características do objeto de estudo e estabelecer relações entre as variáveis. Sua abordagem é, em sua maior parte, qualitativa, pois a maioria dos resultados da pesquisa-ação não pode ser mensurada numericamente e há uma relação dinâmica, entre o objeto de estudo e o pesquisador, que não pode ser traduzida em números" (CARVALHO; MELLO, 2012).

A estratégia de análise consistiu em uma pesquisa com base empírica elaborada em conjunto com uma atividade ou com a correção de um problema enfrentado por pessoas participantes do projeto. A pesquisa teve duração de Janeiro de 2008 a junho de 2009.

Conforme mencionam os autores, a escolha de uma empresa de pequeno porte do ramo de TI com foco em pesquisa e desenvolvimento e que oferece serviços de soluções em tecnologia para diversos clientes. A escolha da B2ML para análise foi motivada pela facilidade de acesso aos dados e às ações que os pesquisadores executam na empresa, o que facilitou a implementação efetiva para experimentação.

Por envolver análises das competências transversais dos times envolvidos no projeto, o método do estudo adotou perguntas extraídas de literaturas gerais como uma tentativa de padronização do método. Mas, claramente, é evidente que padronizar ou ter indicadores para critérios puramente humanos é um desafio para pesquisas metodológicas, o que não invalida de forma alguma a forma escolhida pelos autores.

O método adotado (pesquisa-ação) foi composto por seis fases, elencadas a seguir. Na fase exploratória foram realizados levantamentos para diagnosticar o processo de desenvolvimento de produtos na empresa e também comunicar às equipes de trabalho quanto ao projeto que seria desenvolvido, inicializando-se, também, um projeto piloto. Na primeira fase, foram implementados parcialmente componentes da metodologia Scrum e observados os resultados junto de todo o time. A segunda fase expandiu a implementação dos componentes da metodologia, os erros encontrados foram tratados e pontos possíveis de serem otimizados, de acordo com a observação da equipe, foram também tratados.

1. **Fase exploratória**: Divulgação da proposta da pesquisa-ação para os membros da empresa, que foram entrevistados e apresentados ao objetivo do estudo. Os autores realizaram um levantamento inicial sobre as condições do processo de desenvolvimento dos sistemas existente. O diagnóstico revelou que o método aplicado era informal, não havendo uma normalização da fabricação dos sistemas, o que prejudicava predições de prazo, custo e também o controle dos projetos - portanto, tais pontos foram o foco de aperfeiçoamento.

2. **Planejamento da implementação do Scrum**: Inicialmente, foram realizadas ações com intuito educativo junto dos funcionários. Planejou-se a implementação, que inicialmente foi gradual e iniciou através de um projeto piloto de desenvolvimento de um *software* para gestão de compras entre empresas.

3. **Fase de ação - Primeira iteração**: Definido o produto de desenvolvimento, seis pessoas se organizaram em uma equipe de execução e os papéis dentro da metodologia Scrum foram separados de acordo com a experiência profissional de cada um dos envolvidos. Após a definição da estrutura envolvendo as pessoas, em conjunto foi decidida a primeira interação - implantar parcialmente as práticas ágeis de Scrum.

4. **Fase de ação - Segunda iteração**: Na segunda fase, foram implementadas as práticas por completo e o time começou a trabalhar com maior detalhamento de acordo com os moldes da metodologia e os erros encontrados foram trabalhados.

5. **Fase de ação - Iteração Final**: Consistiu na implementação completa de todos os cargosdo Scrum.

6. **Fase de avaliação**: Análise dos resultados: Pesquisas foram aplicadas para avaliar a evolução do projeto mediante o uso do método. Os resultados dessa fase são mostrados na seção 4.

Na etapa de iteração final, as práticas ágeis de Scrum foram completamente implementadas com maior acuracidade. Por fim, a análise dosresultados foi apresentada. As adequações dos papéis e responsabilidades são mostradas nas Figuras 14, 15 e 16.

Figura 14 – Time: Ações a serem tomadas

1. Time (membros do time)	1.1 Trabalham lado a lado. Este aspecto é importante para melhorar a comunicação e aumentar a sinergia no time
	1.2 Colaboram entre si para implementar os requisitos do *backlog* do *sprint*
	1.3 Têm capacidade de desempenhar vários funções. Deve-se evitar uma superespecialização do time, de modo que um compnente não saiba realizar a tarefa de outros.
	1.4 Admitem quando tem problemas e pedem ajuda ao Scrum master quando isto acontece
	1.5 Ajudam-se mutuamente. Eles precisam entender que as metas do projeto são todas coletivas: Não há vitória ou derrota individual.
	1.6 Aceitam responsabilidades e se comprometem. O time deve se auto gerenciar, o que torna esta característica fundamental.
	1.7 Não necessitam fazer hora extra sistematicamente, permitindoq eu tenham uma vida social agradável e menos estressante.
2. *Scrum master*	2.1 Todos os time têm um
	2.2 Trabalha ao lado de seu time e está presente quando solicitado
	2.3 Têm como prioridade máxima resolver os impedimentos da equipe (Ele trabalha para resolver os itens do backlog dos impedimentos).
3. Dono do produto (*Product Owner*)	3.1 Todos os time têm um. Está sempre disponível para o time tirar dúvidas quanto as regras de negócio
	3.2 Entende o produto e as necessidades do cliente (ou do público alvo do produto) a ponto de tirar dúvidas da equipe sobre seu funcionamento esperado em cada situação
	3.3 Entende o produto e as necessidades do cliente(ou do público alvo do produto) a ponto de saber quais são as funcionalidades prioritárias e, portanto, que devem ser implementadas primeiro.
	3.4 Tem o poder de fazer o time trabalhar primeiramente para implementar as funcionalidades prioritárias.
4. Backlog do produto (*Product Backlog*)	4.1 O dono do produto tem total controle sobre o backlog do produto, podendo alterar prioridades e colocar novos itens.
	4.2 Ele está facilmente acessível e visível para todo o time a qualquer momento.
	4.3 Ele é atualizado após cada sprint, durante a reuniãode revisão do sprint.
	4.4 Conmtém somente funcionalidades do produto e nada mais. O backlog do produto não contém nenhuma tarefa para o time.
	4.5 Cada funcionalidade do backlog do produto contém uma definição própria, de modo que não haverá dúvidas sobre considerá-la ou não implementada.
5. Estimativas	5.1 O dono do produto recebe da equipe estimativas de quantidade de trabalho de cada funcionalidade.
	5.2 O time tem liberdade de estimar e não sofre pressões externas para alterar estimativas.
	5.3 Todos os membros do time participam das estimativas.

Fonte: Adaptado de (CARVALHO; MELLO, 2012)

Figura 15 – Time: Ações a serem tomadas - Continuação

6. Reunião de planejamento do Sprint	6.1 Todos os membros do time participam dela.
	6.2 Ela resulta em um plano do sprint, com um backlog do sprint. Este backlog do sprint está corretamente priorizado de acordo com o dono do produto.
	6.3 Todos os membros do time devem concordar que o planejamento ficou realista. Caso contrário, a reunião deve continuar até se chegar a um consenso.
	6.4 Todos os membros do time se comprometem com o planejado.
7. Sprint	7.1 O time entrega um produto (ou protótipo) funcionando no final de cada sprint.
	7.2 O time segue rigorosamente as prioridades do backlog do sprint.
	7.3 O time age corretivamente quando está atrasado.
	7.4 O time alerta o Scrum master e o dono do produto quando há problemas.
	7.5 O time sabe quando encontrar informações detalhadas que sejam úteis para implementar cada funcionalidade.
	7.6 Os problemas são discutidos e resolvidos no momento em que ocorrem.
	7.7 A duração de cada sprint é sempre a mesma durante um projeto.
	7.8 O intervalo entre 2 sprints é de, no máximo, um dia.
	7.9 Todos os envolvidos (Incluindo clientes e outros times de outros projetos da empresa) sabem sobre o sprint, seus prazos e quais são seus produtos finais.
	7.10 As funcionalidades que começam a ser implementadas num sprint terminam no mesmo sprint.
8. Reunião diária (Daily Scrum)	8.1 Acontecem no mesmo lugar e horário todos os dias
	8.2 Começam e terminam pontualmente.
	8.3 Todos os membros do time estão presentes.
	8.4 Nela, todos os membros do time respondem as 3 perguntas: O que fiz ontem? O que farei hoje? O que está me impedindo de fazer o que é preciso?
	8.5 Nela não acontecem interrupções.
	8.6 O dono do produto a visita reguarlmente.
	8.7 Os membros da equipe buscam as terfas. Cada um decide que tarefa irá fazer. Não é o scrum master quem delega as atribuições.
	8.8 Os membros da equipe cobram entre si a realização das tarefas.

Fonte: Adaptado de (CARVALHO; MELLO, 2012)

Figura 16 – Time: Ações a serem tomadas - Continuação

9. Reunião retrospectiva	9.1 Acontece no final de cada sprint.
	9.2 Todos os membros da equipe e o dono do produto participam.
	9.3 Resulta em sugestões concretas de melhoria.
	9.4 Algumas sugestões apontadas pela reunião retroséctiva são implementadas de fato.
10. Backlog de impedimentos (Impediment backlog)	10.1 Todos os times têm um.
	10.2 Está visível para todos. Além disso, qualquer membro do time pode inserir novos impedimentos a qualquer momento com facilidade.
	10.3 Está sempre atualizado.
	10.4 Tem itens priorizados. O scrum master tem o objetivo de resolvê-los, primeiramente pelos mais importantes.
	10.5 Seus itens que não podem ser resolvidos pelo Scrum master ou pela equipe são destinados para o dono do produto ou para a alta gerência da empresa.O time cobra diariamente pela resolução do problema.
11. Velocidade	11.1 A velocidade de desenvolvimento é mensurada.
	11.2 A velocidade é utilizada para ações corretivas.
	11.3 O registro da velocidade é utilizado para melhorar a estimativa da equipe em planejamentos posteriores.
12. Gráfico burndown	12.1 O time tem um gráfico burndown.
	12.2 O gráfico burndown é visível para toda a equipe.
	12.3 O gráfico burndown é atualizado, no mínimo diaramente pelo scrum master. Preferencialmente, ele é atualizado logo após a implementação de uma funcionalidade.
	12.4 O time age corretivamente caso o gráfico burndown mostre que o desenvolvimento está fora do planejado.
13. Backlog do sprint	13.1 Todos os time tem um.
	13.2 É visível por toda a equipe.
	13.3 É atualizado diariamente (E facilmente) por toda a equipe.
	13.4 A estimativa de trabalho para as tarefas é atualizada diramente.
	13.5 Apesar do backlog do sprint conter funcionalidade e tarefas, elas são facilmente distinguíveis.
	13.6 Nele estão bem claras quais tarefas foram originadas por cada funcionalidade.

Fonte: Adaptado de (CARVALHO; MELLO, 2012)

Os objetivos principais de observação foi de conferir e medir a eficácia das ações adotadas ao longo do projeto no contexto organizacional da empresa e extrair os pontos de melhoria que seriam úteis para expansão da aplicação do Scrum no futuro.

Os autores verificaram que as equipes notaram melhoria significativa na motivação, comunicação e colaboração e houve a redução de prazos e riscos associados ao projeto. Outros benefícios elencados foram a diminuição dos custos com mão de obra e o aumento da produ- tividade da equipe. Como o produto não foi comercializado de fato, não foi possível medir resultados sobre a qualidade do software, mas os autores concluem que a implementação do método Scrum se mostrou adequado para a gestão de projetos de sistemas digitais, apresentando grandes melhorias de trabalho entre as pessoas. A organização tornou-se mais competitiva pelo fato de o projeto ter tido sucesso.

4 CONCLUSÕES

4.1 ESTUDO DE CASO: SCRUM

No estudo de caso, para entender práticas do Scrum, Carvalho e Mello (2012) relatam que na primeira fase de ação, pois a iteração entre as pessoas do time não foi uniforme, havendo falhas de comunicação ou de encontros pessoais na empresa devido à diferença de horários em que os membros estavam disponíveis. Assim sendo, fica nítido que o autogerenciamento de cada indivíduo, ou a ausência dele, impacta diretamente o desenvolvimento da equipe como um todo.

Na segunda etapa, de detalhamento do produto, os autores concluem que a falta de conhecimento técnico específico inerente ao desenvolvimento do *software* em questão, causou impactos negativos e impediu que houvesse essa completa implementação. Isso mostra como é importante ter profissionais com habilidades técnicas compatíveis ao projeto envolvidos no processo. O Scrum *Master* pode, neste contexto, ser ator diferencial no desempenho da equipe, caso tenha as experiências e habilidades técnicas. Os recursos humanos são essenciais à execução da metodologia, conforme ressalta o próprio Manifesto Ágil (2001).

As Reuniões de Planejamento da *Sprint* foram implementadas com sucesso. Entretanto, na etapa de Revisão delas, o time pôde perceber que a falha na definição das entregas. Muitas vezes as atividades priorizadas eram extensas e, ainda que fossem claros para todo o time, houve ausência da estimativa de esforço para cada item do *Backlog* implicou em erros durante as *sprints*. Através dessas conclusões obtidas no estudo de referência, torna-se nítido como o papel do Dono do Produto é essencial pelo fato de auxiliar no entendimento das entregas mais pertinentes à necessidade do cliente, priorizando as entregas e controlando o *Backlog*.

Apesar de serem bem aceito na indústria de produtos digitais e estar em crescente aplicação, o Scrum sofre críticas por alguns autores mais tradicionais, que alegam que o método apresenta falta de escalabilidade para equipes grandes e que trabalham geograficamente distantes e devido a necessidade de mudança de cultura de toda a organização (GREGÓRIO, 2007).

Em termos técnicos de desenvolvimento, a implementação completa do projeto aconteceu na interface "Eclipse 3.2.1"e a codificação gerou 23 pacotes de classes, 89 classes, 893 métodos em classes, 8930 linhas de código na linguagem Java e 14.850 linhas de código em linguagem HTML (*HyperText Markup Language*) - para os padrões da empresa, isso configurou um grande sistema (CARVALHO; MELLO, 2012).

A primeira fase foi avaliada através de pesquisa contendo afirmações sobre benefícios do Scrum extraídas da literatura (CARVALLHO, 2009) e enviadas aos participantes para que votassem. Para analisar a Produtividade e Custos de Produção, outras quatro afirmações foram elaboradas conforme mostra o Quadro 1.

Figura 18 – Afirmações sobre os benefícios Scrum

Item	Afirmações
1	O método Scrum melhorou a comunicação e a colaboração entre os envolvidos.
2	O método Scrum aumentou nossa motivação.
3	O método Scrum facilitou para que o projeto terminasse mais rápido.
4	O método Scrum diminuiu os riscos do projeto e as possibilidades de insucesso.

Fonte: Adaptado de (CARVALHO; MELLO, 2012)

A afirmação que mais apresentou concordância pelos participantes foi a melhoria na comunicação e colaboração entre os integrantes do time. Ainda, conforme concluem os autores, a concordância entre as afirmações 2, 3 e 4 leva à conclusão de que a maioria das pessoas pôde entender as vantagens do método Scrum.

Figura 18 – Média e desvio padrão das respostas

Item	Média	Desvio
1	4,67	0,52
2	4,00	0,63
3	4,17	0,75
4	4,17	0,41

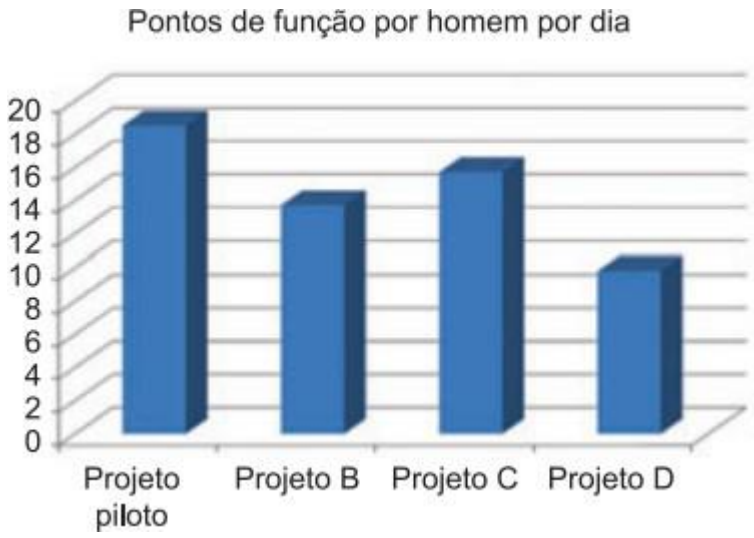
Fonte: Adaptado de (CARVALHO; MELLO, 2012)

Uma análise quantitativa das equipes foi realizada através da comparação do projeto em Scrum com outros três projetos da empresa que estavam em andamento no mesmo período. Os projetos foram chamados de A, B, C e D de forma que o projeto A representa o piloto. A Figura 19 traz os resultados por pontos de função por homem por dia.

Os autores destacam que fatores puderam interferir na produtividade do time, como "efeito Hawthorne"onde eles podem ter trabalhado com mais afinco porque tinham ciência de estarem sendo analisados. Além disso, os projetos tinham times diferentes e isso altera a produtividade.

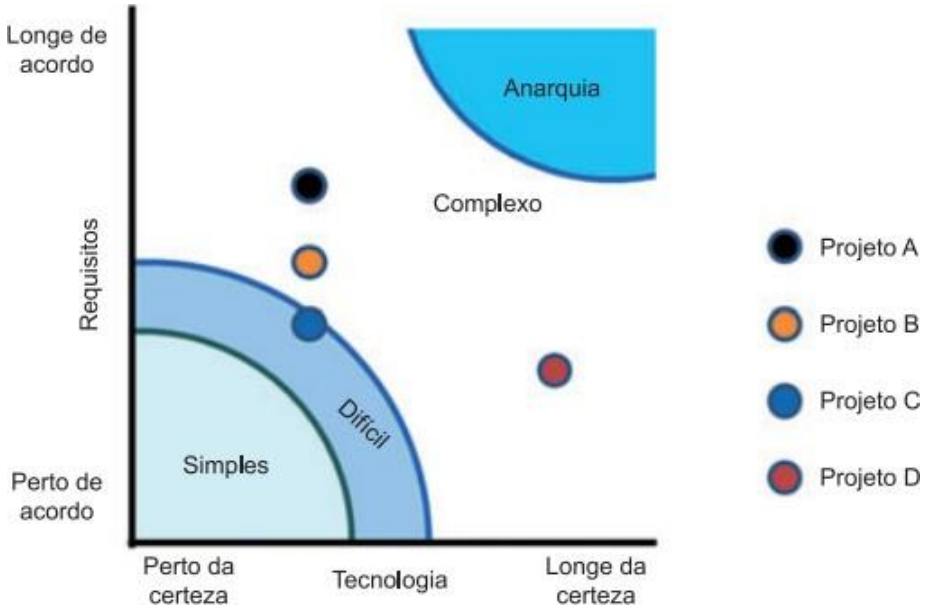
Outro ponto é o nível de dificuldade dos projetos e a Figura 20 traz a classificação dos projetos aplicada ao estudo, seguindo o método aplicado por Schwaber e Beedle (2002) . O piloto A com aplicação do Scrum e os projetos B e C tiveram como base uma tecnologia igualitária, porém os requisitos eram diferentes. O projeto D era único em todos os aspectos. Portanto, afirmar de maneira categórica que o Scrum trouxe melhorias na produtividade do time é inadequado considerando as ponderações apresentadas, cabendo aprofundar os estudos para entender este ponto, mas cabe ressaltar que os dados obtidos são fortes indícios de que houve aumento de produtividade devido a implementação do Scrum.

Figura 19 – Produtividade por projeto



Fonte: (CARVALHO; MELLO, 2012)

Figura 20 – Classificação dos projetos



Fonte: (CARVALHO; MELLO, 2012)

Para aprofundar os estudos, o estudo de referência executou uma segunda etapa de análises que consistiu em aplicar um seminário junto à equipe participante do projeto para discutir e avaliar as práticas do Scrum - vide Figura 21. O time apresentou respostas positivas ao método usado e destacou a melhoria na comunicação e iterações com o cliente, condizente com um processo mais focado em resultados.

Figura 21 – Análises práticas gerenciais do Scrum

Item	Vantagens	Desvantagens
Backlog do Produto	Sua adoção melhorou o planejamento do time	O time sentiu dificuldade para elaborar sua primeira versão que geralmente é muito extensa.
Reunião diária	Sua utilização aumentou muito o controle do projeto. Os riscos foram minimizados.	Houve algumas falhas de periodicidade devido a desencontros de horários dos membros da
Sprint	A divisão do projeto em *Sprints* aumentou a motivação do time.	Devido à inexperiência da equipe, algumas *Sprints* tiveram duração muito longa.
Planejamento da *Sprint*	Sua realização melhorou o planejamento da *Sprint*.	As primeiras reuniões foram pouco produtivas devido à inexperiência da equipe.
Backlog do *Sprint*	Sua boa elaboração é crucial para o sucesso ou fracasso da *Sprint* .	Seu superdimensionamento causou um atraso nas primeiras *Sprints* .
Revisão da *Sprint*	Importante reunião para o recolhimento das lições aprendidas.	Tende a não ser executado pela equipe quando o time está ansioso para o planejamento da
Backlog de Impedimentos	É uma boa ferramente para cobrar a resolução de problemas conhecidos, diminuindo os riscos	Tende a ser negligenciado pelo próprio Scrum *Master* , para evitar maiores responsabilidades.
Velocidades e Estimativas	Muito importante para o controle do projeto e para o planejamento de custos da empresa.	Foi o item mais complexo de ser implementado e que exigiu mais treinamento e novos
Gráfico *burndown*	É uma ferramenta visual interessante, que deixa claro o que os números já mostravam.	Tende a ser abandonado, pois, à primeira vista, parece informação redundante.
Dono do Produto	Facilita para o time, que passa a ter um representante do cliente dentro da própria empresa.	Não foi o caso de projeto estudado, mas provavelmente é difícil encontrar uma pessoa que entenda realmente o negócio do cliente a
Scrum *Master*	É imprescindível para tirar dúvidas quanto ao processo de desenvolvimento e liderar as cerimônias do *Scrum*.	Muitas vezes o time o vê como um gerente do projeto, deixando para ele atribuições que deveriam ser do próprio time.

Fonte: Adaptado de (CARVALHO; MELLO, 2012)

Além disso, outros pontos destacados como sendo importantes para a equipe envolvida no piloto foram levantados em um segundo seminário executado para obter lições aprendidas - vide Figura 22.

Portanto, devido aos resultados apresentados, fica evidente o impacto sentido pelo time do projeto em quatro vertentes do Srum e também pelos dados quantitativos obtidos. Tais pontos seguem:

- Melhoria na comunicação e aumento da motivação do time;
- Redução do tempo investido para finalização do projeto;
- Redução dos riscos de falha (menor probabilidade de insucesso);
- Redução dos curtos com mão de obra;
- Potencialização da produtividade do time;

Em contra partida, a análise da qualidade do *software* em si foi, de fato, limitada devido aos pontos apresentados anteriormente: a falta de comercialização do produto tornou impossível aprofundar tais análises, incluindo a observação da satisfação dos clientes e retorno do investimento.

Figura 22 – Comparativo antes e após a implantação do Scrum

Item	Antes	Depois
Gerência do projeto	Cada time tinha um gerente. Ele tinha como responsabilidades o controle do projeto, o contato com o cliente e assumia seus riscos.	O filme se autogerencia. Todos os membros do time assumem os riscos do projeto e têm comprometimento com seu sucesso.
Contato com o cliente	Somente o gerente do projeto interagia com o cliente.	Criou-se a figura do dono do produto, o qual tem grande interação com o cliente e conhece suas regras de negócio.
Velocidade	Não havia medição formal da velocidade do projeto. O único controle eram os prazos.	A velocidade do projeto é medida e um gráfico de controle visual de toda a equipe é atualizado diariamente.
Planejamento do projeto	O projeto era inteiramente planejado em seu início em detalhes.	O projeto é rapidamente planejado no início e são realizados replanejamentos em cada *Sprint* .
Funcionalidades	As funcionalidades eram implementadas sem uma ordem específica.	Os itens de maior prioridade são implementados antes, de modo que o retorno do investimento ocorra mais rapidamente e com funcionalidades opcionais são implementadas se o tempo permitir.
Lições Aprendidas	Não havia método formal para documentar as lições aprendidas.	O time realiza reuniões de revisão para documentar as lições aprendidas.
Riscos	A gestão de riscos não era uma tarefa formal e ficava sempre a cargo do gerente do projeto.	Existe o *Backlog* de Impedimentos que é uma boa ferramenta para todo o time conhecer os riscos e cobrar a mitigação de riscos conhecidos.
Entregas	São definidas no início do projeto. Geralmente, é feita apenas uma entrega, já do produto final.	As entregas são diversas durante todo o projeto, sempre com um produto com melhorias incrementais em relação ao anterior.

Fonte: (CARVALHO; MELLO, 2012)

O autor do presente estudo conclui, portanto, que a aplicação de métodos ágeis deve ser condicionada à depender das características, não só do projeto, mas também da corporação em que se aplica. E cabe analisar alguns pontos imporantes.

O fato de os meios de gerenciamento serem desburocratizados no agilismo, pode trazer diversas complicações em corporações estruturadas, ou com grande número de funcionários e/ou rotatividade entre eles. A ausência de procedimentos pode criar diversos lapsos de comunicação e perdas no histórico do projeto, fazendo com que definições do *software* possam ser perdidas e, consequentemente, haja retrabalho.

Outro fator é que, pelo fato de não haver a obrigatoriedade da existência de um cronograma oficial do projeto, com os marcos de avanços e descrição clara e completa de todas as subtarefas que serão envolvidas em cada entrega principal, pode fazer com que diversos atrasos ocorram. Além disso, esse mesmo fato pode levar à dificuldades orcamentárias, pois a ausência de marcos (também conhecidos como *milestones*, palavra em inglês de mesmo significado) pode comprometer o pagamento de faturas referentes ao desenvolvimento do projeto. A ausência de cronogramas oficiais pode ser muito prejudicial em projetos grandes, com muitos desenvolvimentos ou equipes terceiras envolvidas.

Também deve-se mencionar que, como ocorre a priorização da entrega mínima de itens de desenvolvidos para o sistema, pode ocorrer muitos erros. Caso ocorra a entrega de um dos itens do *backlog* desenvolvido durante uma *sprint* e o cliente não fique satisfeito, ou, ainda, identifique erros conceituais aplicados ao algoritmo, muito retrabalho não previsto pode ser necessário. Isso faz com que a definição dos recursos envolvidos no projeto não tenha muita acurácia, o que pode causar diversos outros transtornos para uma empresa e sua estimativa orcamentária.

Outro destaque vai para as figuras envolvidas nos projetos, os papeis devem ser muito claros e desenvolvidos, as pessoas devem estar inteiramente envolvidas nas técnicas do método e cientes de suas responsabilidades - o que pode ser um desafio muito grande para corporações antigas e tradicionais que desejam fazer a transição para o agilismo. A ausência de um gerente do projeto que tenha a experiência técnica mínima, tanto nos critérios mínimos exigidos para o sistema, quanto experiência em conceitos de gerenciamento mais aprofundados, com base nos métodos tradicionais que embasam o agilismo, pode trazer complicações para controle dos avanços. Quanto maior o tamanho do projeto e as equipes, maior o risco de ausência de controle utilizando métodos não tradicionais.

Não é incomum ouvir o termo “falso agilismo” quando se atua no ramo empresarial, o que reflete que pelo fato de as metodologias apresentarem rápidos retornos no desenvolvimento de produtos digitais e, ainda, os times envolvidos apresentarem, de modo geral, satisfação em atuar através do método, muitos profissionais e empresas têm alterado sua forma de trabalhar para aplicação dos métodos ágeis, mas isso pode ocorrer de maneira desodernada, dando a falta impressão de que os resultados serão obtidos de maneira satisfatória. Para grandes empresas, o falso agilismo pode ser facilmente encontrado em grupos de trabalho, que muitas vezes podem não estar preparados tecnicamente para mudanças entre as “formas antigas” de gerenciamento e as “formas atuais”.

São diversas as nuances envolvidas na aplicação do agilismo em corporações e os profissionais devem estar atentos ao contexto de aplicação das novas metodologias. Deve existir um cenário culturalmente favorável, com pessoas cientes dos processos e com a dinâmica necessária para responder rapidamente aos desafios. Além disso, a corporação como um todo deve ser capaz de atuar com essa proposta, pois de nada adianta um setor – de projetos, por exemplo – atuar de maneira separada de todos os outros setores, pois as taxas de falha podem ser altas devido ao aspecto cultural e diversos outros temas supracitados.

Conclui-se, por fim, os métodos ágeis são de fato promissores para atender às demandas do mercado de maneira mais rápida no desenvolvimento de programas computacionais – e, portanto, de soluções para o mercado digital. Entretanto, cabe à equipe de gestão fazer análises da pertinência adequada para aplicação dos métodos, que se mostram ser facilmente aplicáveis a pequenas empresas, ou times de desenvolvimento, ou então em cenários preparados para receber a proposta agilista. Entretanto, é imprescindível citar que as técnicas tradicionais de gerenciamento de projetos trazem robustes para o processo, podem facilitar a existência de históricos, que por sua vez pode auxiliar a transição de pessoas envolvidas no desenvolvimento.

5 CONSIDERAÇÕES FINAIS

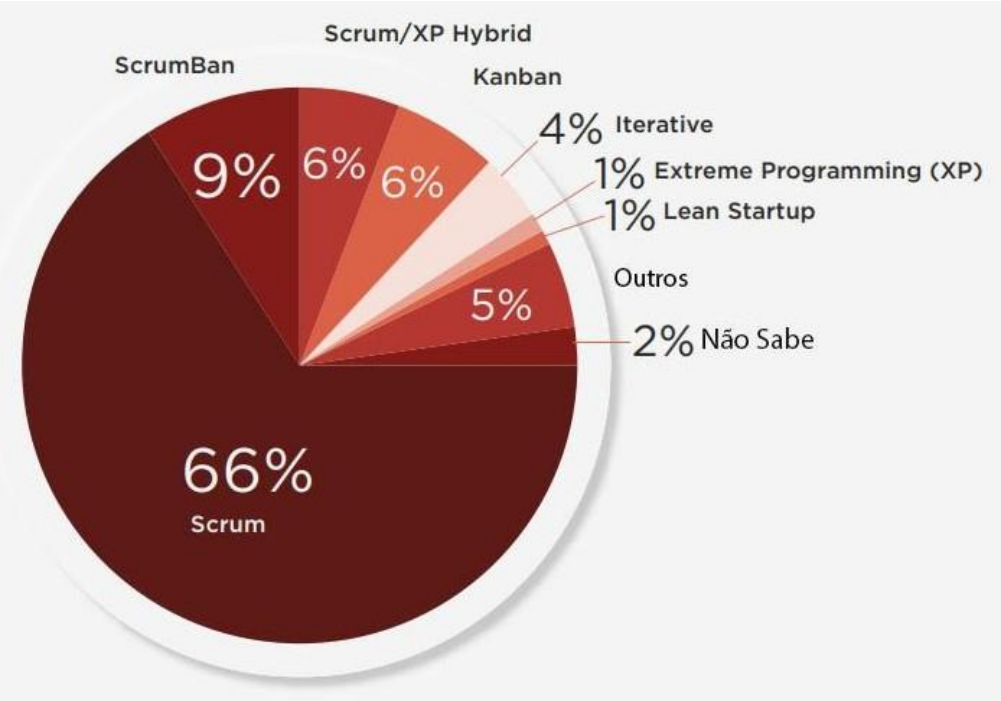
Quando o assunto é o gerenciamento de projetos, sejam digitais ou não, há métodos diferentes de acordo com cada realidade. Por isso, nem sempre os métodos ágeis serão a melhor opção para uma empresa, pois toda a estrutura deve estar preparada para receber os processos e atingir o objetivo final. Existem diversos métodos que podem ser aplicados e cada um será aplicável a uma realidade, ambiente, recursos específicos e cabe ao profissional de engenharia ter ciência de que várias formas de gestão existem e podem ser futuramente pesquisadas. A Figura 23 mostra algumas das outras metodologias ágeis encontradas no mercado.

Quanto aos desafios dos métodos ágeis, o 15º relatório anual elaborado em forma de pesquisa e publicado em 2021 pela State of Agile, mostra dados importantes de acordo com a perspectiva de pessoas que atuam na área e foram entrevistadas. A seguir são elencadas as respostas que surgem da pesquisa quando os participantes apresentam os desafios que permeiam a aplicação dos métodos ágeis (STATE OF AGILE, 2021):

- 46% dos desafios provém de inconsistências no processo e práticas aplicadas pelo time;
- 43% de chances de a estrutura cultural de uma empresa ser um desafio para implementar o agilismo;
- 42% de resistência das organizações às mudanças;
- 42% de falta de habilidades ou experiências com métodos ágeis;
- 41% por não contar com participação da liderança de maneira satisfatória;
- 40% de gerenciamento inadequado e falta de mentorias são motivos de fracassos nos projetos ágeis;
- 35% de treinamentos e conteúdos educacionais são insuficientes;
- 31% da ausência da ponte entre negócios;
- 30% acreditam que ferramentas, dados e medição do projeto fragmentados são desafios a serem vencidos;
- 22% de falta de habilidades de admitir que aconteceram erros e que se deve aprender com o processo;
- 17% de falta de colaboração mínima e a paridade de conhecimentos levam à fracassos;
- 13% acreditam que medidas governamentais e regras como sendo barreiras;
- 7% não souberam responder;
- 5% apontaram outros motivos.

Diversas outras práticas de gestão de projetos de sistemas são aplicadas no mercado, conforme mostra o *15th State of Agile Report* de 2021 e, conforme ilustra a Figura 23, e podem ser estudadas em trabalhos futuros. É interessante notar a fusão de métodos, como o Scrum e Kanban (chamada de Scrumban) - que também podem ser explorados em estudos futuros.

Figura 23 – Metodologias Ágeis mais aplicadas por equipes de desenvolvimento em 2021



Fonte: Adaptado de (STATE OF AGILE, 2021)

De modo geral, é desafiador realizar adaptações nas bases de grandes corporações, por isso torna-se mais fluida a incorporação dos métodos em pequenas empresas, ou empresas de produtos totalmente digitais. Isso também pode ser verificado através do Relatório CHAOS apresentado, onde nota-se que para projetos de tamanho médio e grande ocorrem as maiores diferenças de Sucesso e Fracasso. O cenário agilista é recomendado para pequenas equipes de desenvolvimento, conforme literatura apresentada ao longo do trabalho, e isso se torna evidente pelos resultados obtidos no estudo de caso, onde os autores conseguiram acompanhar de perto o processo de implementação da metodologia visto o pequeno porte da corporação.

Esses dados refletem que a adaptação para o agilismo, embora esteja em alta e sendo procurada por diversas empresas, precisa de toda uma adequação organizacional e cultural, que envolva todas as pessoas no processo. A prática exige uma mudança completa de todo o funcionamento de gestão de uma empresa, portanto interpretações simplistas ou sem embasamento completo podem ocasionar erros.

Isso chama atenção para a definição dos processos de digitalização em si: não basta somente reunir as informações esparsas de uma empresa em uma interface que necessitará de preenchimento manual; apesar de essa solução trazer conforto e até certa rapidez pelo fato de existir um sistema de interface central, ainda haverá desperdício de tempo durante as etapas do

processo de trabalho humano. A real digitalização implica em desenvolver sistemas que sejam capazes de ter autonomia nas tomadas de decisão.

O estudo aqui apresentado tem características multidisciplinares, visto que diversos temas que incorporam a produção de produtos foram mencionados. Como sugestão de trabalhos futuros, a implementação das metodologias ágeis no ambiente de trabalho favorável à tal pode apresentar resultados muito importantes para validação prática dos métodos. Também é possível realizar uma investigação acerca das outras práticas de gestão da implementação de *softwares* a fim de realizar um comparativo entre todas as técnicas.

É de responsabilidade do profissional de engenharia de produção ser capaz de escolher entre as melhores técnicas de gerenciamento que podem ser aplicadas a cada realidade em que se faz presente, tornando-se peça chave para a melhor administração e gestão de empresas. Portanto, o presente trabalho cumpre seu papel ao oferecer subsídios para o entedimento do contexto geral do gerenciamento de projetos de *software*, abordando as metodologias clássicas que compõe o histórico de controle de projetos e também abordando as novas técnicas ágeis que estão crescendo no mercado atual.

REFERÊNCIAS

ABEGGLEN, James C. Kaisha: the Japanese corporation. **The International Executive**, Wiley Online Library, v. 28, 1986. Citado na p. 26.

ALLIANCE, Agile. **Agile Practice Guide, Project Management Institute**, 2017: Agile Practice Guide. [S.l.]: Bukupedia, 2017. v. 1. Citado na p. 27.

AMBLER, Scott. **Agile modeling: effective practices for extreme programming and the unified process**. [S.l.]: John Wiley & Sons, 2002. Citado na p. 9.

ANALISTA EXPERT, website. **Scrum**. [S.l.]. Disponível em: <https://analistaexpert.com.br/scrum/>. Acesso em 29 de julho de 2022, 2020. Citado na p. 34.

ATÉ O MOMENTO, website. **Perguntas e Respostas sobre Agilidade (frameworks, métodos, técnicas, mindset)**. [S.l.]. Disponível em: <https://www.ateomomento.com.br/faq05-perguntas-e-respostas-sobre-agilidade/>. Acesso em 29de julho de 2022., 2019. Citado na p. 30.

BASIL, Victor R; TURNER, Albert J. **Iterative enhancement: A practical technique for software development**. IEEE Transactions on Software Engineering, IEEE, 1975. Citado na p. 20.

BASSI FILHO, Dairton Luiz. **Experiências com desenvolvimento ágil**. São Paulo, 2008. Citadonas pp. 15, 18, 22.

BOAR, Bernard H. **Application prototyping: a requirements definition strategy for the 80s**. [S.l.]: John Wiley & Sons, Inc., 1984. Citado na p. 21.

BOEHM, Barry W. **A spiral model of software development and enhancement**. Computer,IEEE, v. 21, 1988. Citado na p. 14.

BROWN, Shona L; EISENHARDT, Kathleen M. **Product development: Past research, presentfindings, and future directions**. Academy of management review, Academy of ManagementBriarcliff Manor, NY 10510, v. 20, 1995. Citado na p. 14.

CARVALHO, Bernardo Vasconcelos de; MELLO, Carlos Henrique Pereira. **Aplicação do método ágil scrum no desenvolvimento de produtos de software em uma pequena empresa debase tecnológica.** Gestão & Produção, SciELO Brasil, v. 19, 2012. Citado nas pp. 40–43, 45–49.

CARVALLHO, Bernardo Vasconcelos de. **Aplicação do Método Ágil SCRUM na gestão de desenvolvimento de produtos de software por uma pequena empresa de base tecnológica**, 2009.Citado na p. 45.

CLANCY, Tom. **The chaos report**. The Standish Group, 1995. Citado na p. 16.

COHN, Mike. **Introduction to Scrum PPT.** [S.l.]. Disponível em <https://www.mountaingoatsoftware.com/agile/scrum/resources/a-reusable-scrum-presentationf,2018>. Acesso em 16 de julho de 2022. Citado na p. 34.

COMPANY, Bain &. **Technology Report 2021**. [S.l.]. Disponível em: <https://www.bain.com/globalassets/noindex/2021/bainreportt echnology − report − 2021.pdf> , 2021. Acesso em 15 de julho de 2022. Citado na p. 15.

FLOYD, Christiane. **A systematic look at prototyping**. In: APPROACHES to prototyping. [S.l.]: Springer, 1984. Citado na p. 20.

GRAVES, ROBERT J; KONOPKA, JOHN M; MILNE, R JOHN. **Literature review of materialflow control mechanisms**. Production Planning & Control, Taylor & Francis, v. 6, 1995. Citado na p. 35.

GREGÓRIO, Matheus. **Os sete pecados na aplicação de processos de software**. UniãoBrasileira dos Institutos de Tecnologia, UNIBRATEC, 2007. Citado na p. 45.

KERR, James; HUNTER, Richard. **Inside RAD: How to build fully functional computer systems in 90 days or less**. [S.l.]: McGraw-Hill, Inc., 1994. Citado na p. 22.

KRAUSHAAR, James M; SHIRLAND, Larry E. **A prototyping method for applications development by end users and information systems specialists**. MIS Quarterly, JSTOR, 1985. Citado na p. 20.

LIKER, Jeffrey K. **Toyota way: 14 management principles from the world's greatest manufacturer**. [S.l.]: Mc Graw-Hill Education, 2004. Citado na p. 35.

OHNO, Taichi. **Just-In-Time for Today and Tomorrow.** Productivity Press, 1988. Citado naspp. 26, 35.

POPPENDIECK, Mary; POPPENDIECK, Tom. **Lean software development: an agile toolkit**. [S.l.]: Addison-Wesley, 2003. Citado na p. 23.

PRESSMAN, Roger S. **Engenharia de Software-7**. [S.l.]: Amgh Editora, 2010. Citado na p. 21.

PRESSMAN, Roger S; MAXIM, Bruce R. **Engenharia de software-9**. [S.l.]: McGraw Hill Brasil, 2021. Citado nas pp. 14, 20, 22.

PRIKLADNICKI, Rafael; WILLI, Renato; MILANI, Fabiano. **Métodos ágeis para desenvolvimento de software**. [S.l.]: Bookman Editora, 2014. Citado nas pp. 10, 18, 28.

QUINTINO, Luıs Fernando et al. **AUTOMAÇÃO ROBÓTICA EM INDÚSTRIAS– EVOLUÇÃO TECNOLÓGICA E COMPETITIVIDADE**. Citado na p. 9.

RISING LINDA E JANOFF, Norman S**. The Scrum software development process for smallteams**. IEEE software, IEEE, v. 17, 2000. Citado na p. 27.

ROYCE, Winston W. **Managing the development of large software systems: concepts and techniques**. In: PROCEEDINGS of the 9th international conference on Software Engineering.[S.l.: s.n.], 1987. Citado na p. 18.

SANTOS SOARES, Michel dos**. Comparação entre metodologias Ágeis e tradicionais para o desenvolvimento de software**. INFOCOMP Journal of Computer Science, v. 3, 2004. Citadona p. 19.

SAWYER, Steve; GUINAN, Patricia J. **Software development: Processes and performance.** IBM systems journal, IBM, v. 37, 1998. Citado na p. 11.

SCHONBERGER, Richard J. **World class manufacturing**. [S.l.]: Simon e Schuster, 2008. Citado na p. 26.

SCHWABER, Ken. **Scrum development process**. In: BUSINESS object design andimplementation. [S.l.]: Springer, 1997. Citado na p. 31.

SCHWABER, Ken; BEEDLE, Mike. **Agile software development with scrum. Series in agile software development**. [S.l.]: Prentice Hall Upper Saddle River, 2002. v. 1. Citado na p. 31.

SCHWABER, Ken; SUTHERLAND, Jeff. **Guia do Scrum–Um guia definitivo para o Scrum: As regras do jogo**. 2013. Disponível em <https://scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf.>. Acesso em 15 de julho de 2022. Citado nas pp. 32, 33.

SHULL, Forrest et al. **What we have learned about fighting defects**. In: IEEE. PROCEEDING. Seighth IEEE symposium on software metrics. [S.l.], 2002. Citado na p. 20.

SINDICONT, website. **Kanban: como utilizar para gerenciar diversos projetos**. [S.l.]. Disponível em <http://www.sindicontitajai.org.br/noticias-empresariais/kanban-como-utilizar-para-gerenciar-diversos-projetos>. 2018. Acesso em 29 de julho de 2022. Citado na p. 37.

STANDISH GROUP, Standish Group. **Chaos Report**. [S.l.]. Disponível em <https://www.infoq.com/articles/standish-chaos-2015>. 2015. Citado na p. 16.

STATE OF AGILE, State of Agile. **15th State of Agile Report**. [S.l.]. Disponível em <https://info.digital.ai/rs/981-LQX-968/images/SOA15.pdf, 2021>. Acesso em 25 de junho de 2022. Citado nas pp. 50, 51.

TAKEUCHI H. E NONAKA, I. **The New New Product Development Game**. [S.l.]: Harvard Business Review, 1986. Citado na p. 30.

VARGAS, Letıcia Marques. **Gerenciamento Ágil de Projetos em Desenvolvimento de Software:um estudo comparativo sobre a aplicabilidade do Scrum em conjunto com PMBOK**. Revista de Gestão e Projetos, v. 7, 2016. Citado na p. 10.

WIDMAN, Jeff; HUA, Stella Y; ROSS, Steven C. **Applying lean principles in software development process–a case study**. Issues in Information Systems, v. 9, 2010. Citadonas pp. 36, 38.

WOMACK, James P; JONES, Daniel T. **From lean production to lean enterprise**. Harvard business review, Harvard Business School Publication Corp., v. 72, 1994. Citado na p. 26.

_____. Lean thinking—banish waste and create wealth in your corporation. **Journal of the Operational Research Society**, Taylor & Francis, v. 48, 1997. Citado nas pp. 23, 26.

WOMACK, James P; JONES, Daniel T; ROOS, Daniel. **The machine that changes the world**. Rawson Associates, NY, 1990. Citado nas pp. 23, 26.

ANEXO E – TERMO DE AUTENTICIDADE

UNIVERSIDADE FEDERAL DE JUIZ DE FORA
FACULDADE DE ENGENHARIA

Termo de Declaração de Autenticidade de Autoria

Declaro, sob as penas da lei e para os devidos fins, junto à Universidade Federal de Juiz de Fora, que meu Trabalho de Conclusão de Curso do Curso de Graduação em Engenharia de Produção é original, de minha única e exclusiva autoria. E não se trata de cópia integral ou parcial de textos e trabalhos de autoria de outrem, seja em formato de papel, eletrônico, digital, áudio-visual ou qualquer outro meio.
Declaro ainda ter total conhecimento e compreensão do que é considerado plágio, não apenas a cópia integral do trabalho, mas também de parte dele, inclusive de artigos e/ou parágrafos, sem citação do autor ou de sua fonte.
Declaro, por fim, ter total conhecimento e compreensão das punições decorrentes da prática de plágio, através das sanções civis previstas na lei do direito autoral[1] e criminais previstas no Código Penal[2], além das cominações administrativas e acadêmicas que poderão resultar em reprovação no Trabalho de Conclusão de Curso.

Juiz de Fora, 09 de Agosto de 2022.

Lucas Franco Ferreira da Silva	201349031
NOME LEGÍVEL DO ALUNO (A)	Matrícula
Lucas F. F. da Silva	08995245670
ASSINATURA	CPF

[1] LEI N° 9.610, DE 19 DE FEVEREIRO DE 1998. Altera, atualiza e consolida a legislação sobre direitos autorais e dá outras providências.
[2] Art. 184. Violar direitos de autor e os que lhe são conexos: Pena – detenção, de 3 (três) meses a 1 (um) ano, ou multa.