UNIVERSIDADE FEDERAL FLUMINENSE
INSTITUTO DE COMPUTAÇÃO
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

BEATRIZ LOPES SOUZA

METODOLOGIAS ÁGEIS: ANÁLISE E COMPARAÇÃO DO *SCRUM, KANBAN* E *LEAN* APLICADOS AO DESENVOLVIMENTO DE SOFTWARE

Niterói
2021

BEATRIZ LOPES SOUZA

METODOLOGIAS ÁGEIS: ANÁLISE E COMPARAÇÃO DO *SCRUM*, *KANBAN* E *LEAN* APLICADOS AO DESENVOLVIMENTO DE SOFTWARE

Trabalho de conclusão de curso apresentado ao curso de Bacharelado em Sistemas de Informação, como requisito parcial para conclusão do curso.

Orientador:
Prof. Dr. Rodrigo Salvador Monteiro.

Niterói
2021

S719m Souza, Beatriz Lopes
Metodologias Ágeis: Análise e Comparação do Scrum, Kanban e Lean Aplicados ao Desenvolvimento de Software / Beatriz Lopes Souza ; Rodrigo Salvador Monteiro, orientador. Niterói, 2021.
82 f. : il.

Trabalho de Conclusão de Curso (Graduação em Sistemas de Informação)-Universidade Federal Fluminense, Instituto de Computação, Niterói, 2021.

1. Engenharia de software. 2. Gerenciamento de software. 3. Produção intelectual. I. Monteiro, Rodrigo Salvador, orientador. II. Universidade Federal Fluminense. Instituto de Computação. III. Titulo.

CDD -

BEATRIZ LOPES SOUZA

METODOLOGIAS ÁGEIS: ANÁLISE E COMPARAÇÃO DO *SCRUM*, *KANBAN* E *LEAN* APLICADOS AO DESENVOLVIMENTO DE SOFTWARE

Trabalho de conclusão de curso apresentado ao curso de Bacharelado em Sistemas de Informação, como requisito parcial para conclusão do curso.

Aprovada em 04 de maio de 2021.

BANCA EXAMINADORA

_____

Prof. Dr. Rodrigo Salvador Monteiro (Orientador) - UFF

_____

Profª. Drª. Vânia de Oliveira Neves - UFF

_____

Prof. Dr. Flávio Luiz Seixas - UFF

Niterói
2021

Especialmente aos meus dedicados pais.

AGRADECIMENTOS

Ao professor Rodrigo Salvador Monteiro por ter aceitado acompanhar-me neste projeto. A sua experiência e parceria foram essenciais para a minha motivação à medida que as dificuldades iam surgindo ao longo do percurso.

Aos meus pais Paulo e Alvanir que sempre me amaram, acreditaram e apoiaram meus sonhos.

A minha irmã Michelle que torceu incansavelmente pela conclusão deste trabalho.

Ao meu namorado Ítalo que foi tão presente nos momentos finais me incentivando a seguir em frente.

As minhas amigas Amanda, Anna Luiza e Millene que estiveram ao meu lado ao longo de todo o período de tempo em que me dediquei a este trabalho.

Ainda há muitas causas pelas quais vale a pena se sacrificar e muita história para ser escrita.

Michelle Obama

RESUMO

A Tecnologia da Informação alcançou um novo patamar mundial na atualidade se tornando um fator crucial para sobrevivência e inovação de empresas do mundo todo. Diante da importância dessa área para os negócios, empresas de tecnologia precisaram pensar em novas formas de desenvolver software de qualidade em tempos mais curtos. Diversas abordagens surgiram para gerenciar a produção de sistemas, entretanto, as metodologias ágeis foram as que ganharam mais força, graças a seus resultados expressivos na gerência de projetos, aumento da qualidade e na satisfação dos clientes. A adoção da agilidade foi capaz de originar novos modelos de trabalho para desenvolver sistemas, dentre os modelos ágeis existentes, *Scrum*, *Kanban* e *Lean* são os mais populares. O objetivo do presente estudo é analisar os modelos ágeis selecionados entendendo suas características e contribuições para a Tecnologia da Informação. O estudo então apresenta um detalhamento dos modelos escolhidos seguido de uma análise comparativa dos mesmos. Como resultado, foi obtido um mapeamento das forças e fraquezas desses modelos e quais são os critérios que devem ser levados em consideração no momento da implantação e uso dessas abordagens nas organizações.

Palavras-chave: Metodologias Ágeis. *Scrum*. *Kanban*. *Lean*.

ABSTRACT

Information Technology has reached a new world level, becoming a crucial factor for survival and innovation of companies worldwide. Given the importance of this area for business, technology companies needed to think of new ways to develop quality software in shorter times. Several approaches have emerged to manage the production of systems, however, agile methodologies have gained the most strength, thanks to their expressive results in project management, increased quality and customer satisfaction. The adoption of agility was able to originate new working models to develop systems, among the existing agile models, Scrum, Kanban and Lean are the most popular. The objective of the present study is to analyze the selected agile models, understanding their characteristics and contributions to Information Technology. The study then presents a breakdown of the chosen models followed by a comparative analysis of them. As a result, a mapping of the strengths and weaknesses of these models was obtained and what are the criteria that must be considered when implementing and using these approaches in organizations.

Keywords: Agile Methodologies. Scrum. Kanban. Lean.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface* - Interfaces de Programação de Aplicações
JIT	*Just In Time*
ROI	*Return over Investment* - Retorno sobre Investimento
RUP	*Rational Unified Process* - Processo Unificado da Rational
STP	Sistema Toyota de Produção
TI	Tecnologia da Informação
WIP	*Work In Progress* - Trabalho em Progresso

SUMÁRIO

1 **INTRODUÇÃO**

É possível entender o conceito de simbiose como um vínculo, a longo prazo, entre dois organismos de espécies diferentes. Durante seu período de existência a humanidade estabeleceu diversas relações simbióticas, de forma a garantir a sua existência e evolução. Curiosamente, é possível considerar que a maior e potencialmente mais irreversível simbiose da história da humanidade, não seja com um organismo vivo, mas sim, com a Tecnologia da Informação (TI).

A informatização do quotidiano humano é um processo diário e crescente a nível global. Sendo assim, não apenas a vida pessoal do indivíduo foi impactada, as corporações também foram capturadas de tal forma por esse processo tecnológico que, de acordo com Domingues (2004), a TI abandonou a posição de um simples coadjuvante no ambiente organizacional e assumiu o papel vital para obtenção do sucesso na estratégia do negócio.

A medida que a TI assume a posição de item de sobrevivência nas empresas, a necessidade de criar e consumir softwares cresce ao mesmo passo. Ao longo dos anos, surgiram diversas organizações e indústrias para suprir a demanda tecnológica do mercado. Com o aumento da oferta de fornecedores, foi necessário não só produzir mais soluções de software, mas também com mais qualidade, custos baixos e prazos competitivos (SAVOINE et al., 2009).

Deste modo, na década de 60 surge a Engenharia de Software e as primeiras metodologias de desenvolvimento de software, na tentativa de estruturar as atividades de desenvolvimento (SIQUEIRA, 2007). As abordagens iniciais eram fortemente ancoradas em planejamento e documentação extensiva, e apesar da tentativa de aprimorar o processo de desenvolvimento de software, elas tiveram dificuldades de otimizar as entregas, gerando problemas como perdas de prazo e orçamento (STANDISH GROUP Inc., 2001).

A partir dos anos 90, no contexto da Engenharia de Software, começou a emergir uma nova forma de trabalho, Metodologias Ágeis. As metodologias e modelos pertencentes a esse novo conceito, foram criadas para mitigar os problemas de gerenciamento existentes nos modelos tradicionais. Desde então, os modelos ágeis evoluíram e foram capazes de se adaptar para gerenciar produtos complexos, a vista disso, tornaram-se um padrão de trabalho sendo largamente adotados.

1.1 **Motivação**

Segundo Herbsleb e Moitra (HERBSLEB & MOITRA, 2001), para as organizações que almejam sucesso, é indispensável o uso da Tecnologia da Informação como diferencial competitivo. A TI se tornou a chave capaz de converter uma organização em uma potência, desse modo, a aquisição e uso de soluções tecnológicas tornou-se substancial para a sobrevivência das instituições, como consequência o desenvolvimento de sistemas conquistou um espaço expressivo no mercado mundial.

A multiplicidade de conexões e decisões que os sistemas computacionais precisam suportar dentro de uma corporação, somado à necessidade que as organizações têm de se manterem competitivas e aderentes às transformações do mercado, exige que os softwares consumidos por elas acompanhem com proximidade a evolução dos negócios. A combinação desses fatores é que torna a atividade de desenvolver sistemas complexa, sensível e mutável, características que eram difíceis de serem dominadas pelas metodologias tradicionais.

Conforme os modelos clássicos acumulavam críticas, os modelos ágeis ganharam força se apresentando como uma saída para lidar com as mudanças. Elas possuem as mesmas preocupações com custo, prazo e qualidade que suas antecessoras, contudo, dispõem de mecanismos que são capazes de se adaptar às flutuações das requisições dos negócios. As metodologias ágeis se consolidaram com um modelo de trabalho para as companhias de TI, em escala mundial.

Diante de todo contexto apresentado anteriormente, este trabalho busca aprender sobre o surgimento, popularização e contribuições dos Modelos Ágeis. No entanto, levando em consideração o grande número de modelos existentes para se trabalhar com agilidade e com o objetivo de enriquecer a discussão, este estudo será focado nos modelos ágeis *Scrum, Kanban* e *Lean* explorando suas aplicações e estímulos no cenário de desenvolvimento de software. Os modelos citados foram escolhidos devido às suas altas taxas de aceitação e utilização pelas organizações.

1.2 **Objetivos Gerais e Específicos**

Nesta seção, apresentam-se o objetivo geral e específicos deste trabalho de conclusão de curso. Como objetivo geral, este trabalho almeja apresentar uma análise sobre metodologias ágeis com foco em examinar os modelos *Scrum*, *Kanban* e *Lean*. Para alcançar tal resultado, o estudo abordará os objetivos específicos:

- Examinar o conceito de metodologias ágeis a fim de compreender histórico e características;
- Discorrer sobre origem, motivação, estrutura e contribuições dos modelos selecionados;
- Analisar comparativamente os modelos ágeis selecionados.

1.3 Organização do Trabalho

Este trabalho está divido em cinco capítulos. No primeiro capítulo é descrito as motivações e considerações iniciais sobre o tema, além dos objetivos gerais e específicos e a organização do trabalho.

No segundo capítulo são detalhadas questões históricas e características das metodologias de desenvolvimento de software tradicionais e ágeis.

O terceiro capítulo é composto por uma apresentação das modelos ágeis de desenvolvimento de software: *Scrum*, *Kanban* e *Lean*. Durante o capítulo são apresentados origem, motivação, estrutura e contribuições desses modelos para a construção de sistemas.

O quarto capítulo expõe uma análise comparativa dos modelos selecionados em relação a critérios significativos nas esferas de Agilidade e Negócio. São apresentadas vantagens e desvantagens dos modelos em relação ao atendimento dos critérios.

O quinto e último capítulo do trabalho apresenta as conclusões acerca da realização do trabalho e sugestões sobre trabalhos futuros.

2 **METODOLOGIAS DE DESENVOLVIMENTO** SOFTWARE

Neste capítulo será apresentada a definição de metodologias de desenvolvimento de software e modelos de software, a partir desse entendimento, serão abordados os dois tipos de metodologias proeminentes no universo de desenvolvimento de sistemas: as metodologias tradicionais e ágeis. Acerca dos dois tipos de metodologia, serão apresentadas as origens e características de cada. Por fim, será explorado com maior profundidade o domínio das metodologias ágeis, uma vez que, estão contidas nelas os modelos de desenvolvimento de software *Scrum*, *Kanban* e *Lean* que são alvo de estudo deste trabalho.

O termo metodologia refere-se a um campo no qual é realizado o estudo dos melhores métodos praticados em uma área de conhecimento. As Metodologias englobam conceitos práticos e filosóficos para conduzir o caminho rumo aos resultados esperados. No âmbito da Tecnologia da Informação, as metodologias guiam pessoas e organizações no desenvolvimento de soluções tecnológicas. A Metodologia define o processo a ser executado, logo, os recursos e abordagens adotados durante o desenvolvimento de sistemas fazem parte de um processo apoiado em uma metodologia.

2.1 **Processos de** Software

Um processo de desenvolvimento de software é um agrupamento de atividades, estruturadas e relacionadas entre si, com o propósito de construir um artefato de software de qualidade (SOMMERVILLE, 2011). Os processos de desenvolvimento fazem parte e são estudados dentro do campo da Engenharia de Software, eles são considerados um importante caminho a ser seguido quando se trata de cumprir com os contratos do desenvolvimento e obtenção de software de qualidade. Há diversos processos de desenvolvimento, contudo todos devem possuir quatro atividades fundamentais para a Engenharia de Software (SOMMERVILLE, 2011):

- Especificação: Definição das funcionalidades e restrições do software.

Projeto e implementação de software: Produção do software para atender a especificação.

- Validação de software: Garantia do atendimento às necessidades para as quais foi construído.
- Evolução de software: Capacidade de evoluir para atender novas demandas.

Essas atividades estão presentes em todos os processos de software de alguma forma, contudo, os mesmos são complexos e, como todos os processos, intelectuais e criativos, dependem do fator humano, ficando expostos a julgamentos subjetivos (SOMMERVILLE, 2011). Diante disso, os processos além de buscarem aderência aos requisitos da Engenharia de Software, também evoluíram no sentido de se adaptarem às condições do ambiente ao qual estão expostos e tirarem proveito dos recursos disponíveis. Logo, é possível compreender que as mudanças nos processos de software estão intimamente ligadas às mudanças da Tecnologia da Informação, da mentalidade das pessoas que trabalham com ela e do mundo.

2.1.1 Modelos de Processos de Software

Os processos de desenvolvimento de software são representados e organizados por diferentes modelos. Um modelo de processo de software é uma representação simplificada de um processo de software, cada modelo representa uma visão particularizada e parcial de um processo, podendo omitir determinados aspectos, como por exemplo, papéis, ferramentas, dependências ou relacionamentos (SOMMERVILLE, 2011). Os modelos não são descritivos definitivos dos processos, eles podem ser vistos como *frameworks* de processo, sendo assim possuem a plasticidade de serem ampliados e adaptados para gerar processos de engenharia de software mais específicos

2.2 **Metodologias Tradicionais**

As Metodologias Tradicionais ou Clássicas surgiram em um contexto de desenvolvimento de software, baseado apenas em um mainframe e terminais burros (ROYCE, 1970). Elas também são chamadas de pesadas ou orientadas a documentação devido ao forte

acoplamento à atividade de planejar e documentar todo o processo. Essa característica deve-se ao fato de que quando foram concebidas, os recursos para codificar e depurar eram limitados, em consequência havia um alto custo para correção, portanto, fazia-se necessário evidenciar detalhadamente toda a lógica pensada para solução a fim de favorecer o desenvolvimento e futura manutenção (LUDVIG & REINERT, 2007).

O modelo cascata, representado na figura 1, foi bem popular dentro das metodologias tradicionais. A característica dominante do cascata, é a existência de uma estrutura sequencial para realização das atividades, onde uma etapa deveria começar logo após a outra, esse sistema apresenta baixa flexibilidade e a impossibilidade de retorno a um ponto anterior da cadeia. Amplamente adotado na época do seu surgimento e ainda pro anos depois, esse modelo é representado por um encadeamento de etapas no qual o processo flui como uma cascata. Ocasionalmente, pode existir uma retroalimentação de uma etapa para a etapa antecessora, entretanto, de um ponto de vista macro, as etapas seguem essencialmente de forma sequencial.

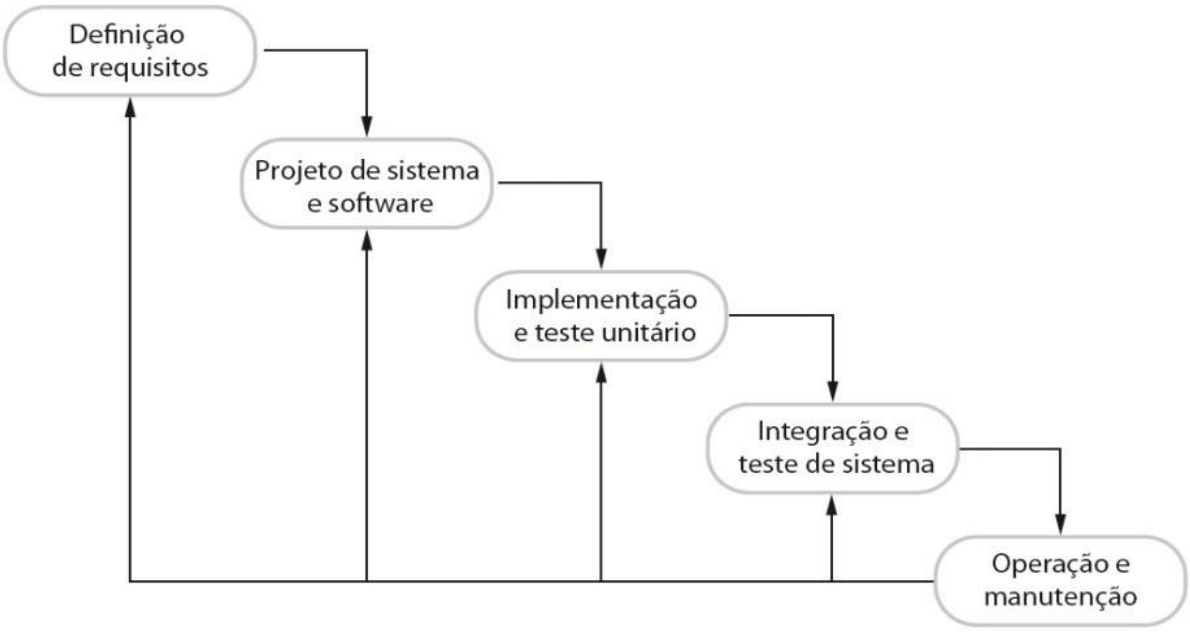

Figura 1 - Etapas de desenvolvimento do modelo cascata. (Fonte: SOMMERVILLE, 2011)

2.2.1 Características das Metodologias Tradicionais

No modelo cascata, é esperado que a execução saia conforme o planejado. Além da ordenação das etapas, é necessário que o planejamento garanta que o fim do projeto atenda o prazo, orçamento e qualidade definida no início. Dessa forma, apenas no fim do processo é que o sucesso e valor do produto pode ser medidos. Caso o cliente venha a solicitar uma mudança no sistema após iniciado o desenvolvimento, tais alterações provocam transtornos nesse processo, ao passo que será preciso revisitar o planejamento do projeto, agregando mais custos, uma vez que se trata de uma forma de trabalho rígida (MARQUES, 2012).

Este formato de desenvolvimento preocupa-se em manter o planejado a fim de entregar o acordado, contudo o nível de detalhamento e descrição das atividades consome uma grande parcela de tempo, ocasionando prazos longos. Logo, era comum ao entregar o produto, o mesmo não ser mais aderente às necessidades do cliente.

A partir do modelo cascata, novos modelos de software começaram a surgir buscando incorporar maior iteratividade no desenvolvimento, como os modelos prototipagem, espiral, RUP (*Rational Unified Process*), dentro outros. Contudo, à medida que o mundo se tornava mais globalizado e as empresas se tornavam mais competitivas, surgiu a necessidade de aprimorar os modelos de trabalho. Dessa maneira, uma nova mentalidade para se desenvolver software ganhou força, para fazer frente às falhas deixadas pelos modelos anteriores, esses novos pensamentos materializam-se nas hoje conhecidas como Metodologias Ágeis. A principal diferença entre as Metodologias Tradicionais das Ágeis, são os valores e princípios que a segunda trouxe para o universo de desenvolvimento de software e examinaremos a seguir.

2.3 **Metodologias Ágeis**

Antes de entender o que são as metodologias ágeis é preciso compreender o conceito de agilidade. Na década de 90, começaram a surgir novas abordagens para lidar com o desenvolvimento de software chamadas de “processos leves”, que tinham como objetivo desburocratizar os processos de desenvolvimento de software tradicionais. Estes são caracterizados pela formalização exagerada nas documentações e regulamentações, em

contrapartida, os “leves” que emergiram, como *Scrum, Extreme Programming* e *Feature Driven Development,* prometiam priorizar o software em si, diminuindo a parte da documentação e buscando entregas mais rápidas.

A maior parte dos projetos até o final dos anos 90 eram gerenciados através de metodologias tradicionais, nas quais o foco era no planejamento que suportava todo o progresso posterior (TOMÁS, 2009). Todavia, conforme a popularidade dos processos leves crescia, um grupo de profissionais de grande referência no mundo de desenvolvimento de software, motivados por esse cenário de insatisfação, em fevereiro de 2001, reuniu-se para discutir melhores práticas para desenvolver seus projetos. Desse encontro, originou-se o manifesto ágil (BECK et al., 2001), uma declaração com valores ágeis e princípios que orientam o desenvolvimento ágil.

A essência da agilidade é lançar sobre o desenvolvimento de software um novo olhar, orientado a adaptação, habilidades comunicativas e capacidade de ofertar novos produtos e serviços de valor ao mercado, em breves períodos de tempo (KOSCHEVIC, 2001). Características essas que podem ser encontradas nos valores e princípios da agilidade descritos no manifesto ágil.

2.3.1 Valores e Princípios Ágeis

Os valores e princípios ágeis, ou simplesmente agilidade, são o cerne do desenvolvimento ágil, mais que metodologias ou modelos, compreender e respeitar essas crenças é o que garante que a agilidade seja absorvida de forma adequada nas empresas, pessoas e projetos. Os quatro valores ágeis são:

- **Indivíduos e interações mais que processos e ferramentas**: Para entender o real significado de “Indivíduos e interações mais que processos e ferramentas”, o processo de desenvolvimento de software é formado por um conjunto de passos relacionados a artefatos, pessoas, estruturas organizacionais e restrições (Wazlawick, 2013). Esse pensamento inclui as pessoas como parte do desenvolvimento e de fato os profissionais envolvidos são fundamentais, uma vez que, eles irão mapear e interpretar as regras e requisitos e transformá-los em código para criar um software utilizável. Diferente dos

modelos tradicionais que traziam regras e acordos para o centro do desenvolvimento, a agilidade traz as pessoas, como forma de abraçar que os softwares são feitos por pessoas e para pessoas.

Explorando ainda mais esse valor, a agilidade preza que as pessoas certas, com habilidades corretas estejam envolvidas no desenvolvimento e que a comunicação entre elas e com as partes interessadas seja eficiente a fim de construir o produto certo. Vale ressaltar que as ferramentas e processos utilizados possuem sua relevância no sucesso do projeto, mas não devem ser mais significativas que a busca pela qualidade através do alto nível de interação entre os envolvidos (UTIDA, 2012).

- **Software em funcionamento mais que documentação abrangente**: Na visão do Manifesto Ágil (2020), as metodologias ágeis priorizam um software funcional ao invés da elaboração de uma documentação extensa, o ponto chave é que o esforço e tempo gasto na produção de uma documentação que não agrega valor ao projeto deve ser direcionado para a atividades de desenvolvimento do software, ou seja, o propósito é evitar a criação de qualquer documentação que não será utilizada, e por isso, o foco precisa estar no software gerado e que ele atenda os requisitos esperados. Contudo, isso não significa que não deve haver documentação em projetos ágeis, mas que toda a documentação criada seja significativa e sincronizada com o projeto.
- **Colaboração com o cliente mais que negociação de contratos**: A agilidade defende que para gerar resultados aderente a necessidade do cliente, é primordial que exista um *feedback* constante (Manifesto Ágil, 2020). Nessa abordagem, os contratos devem apoiar o projeto definindo questões mais abrangentes como comunicação e relacionamento entre as partes, deixando com maior flexibilidade fatores como custo, escopo e prazo que podem flutuar de acordo com o recebimento de novos inputs do negócio e mercado. Diferente das metodologias tradicionais que eram rígidas em relação ao que foi acordado inicialmente, a agilidade se preocupa em entregar valor mesmo que isso signifique ter alterações nos acordos iniciais.

Para que o ciclo de *feedback* funcione em tempo plausível, os incrementos nos projetos ágeis são pequenos e disponibilizados em ciclos curtos de tempo e entregues constantemente ao cliente de forma a envolvê-lo no processo de construção de um produto adequado. Essa forma de interação, também auxilia na minimização da documentação, pois existe uma constante comunicação entre os indivíduos interessados no sistema (KOSCHEVIC, 2001).

- **Responder a mudanças mais que seguir um plano**: A agilidade é motivada a se adaptar mais do que seguir à risca o que foi planejado, pois é certo que mudanças irão ocorrer, então é preciso apoiar-se em processos e ferramentas que as suportem de forma eficiente. No entanto, o planejamento deve acontecer, mas para períodos reduzidos (UTIDA, 2012), curtos o suficiente para mitigar a maior parte dos riscos relacionados à incerteza e ser possível corrigir os desvios em tempo que não prejudique o desenvolvimento.

De acordo com Marçal (et. al, 2011), como agilidade deve-se entender "a habilidade de criar e responder a mudanças, buscando a obtenção de lucro em um ambiente de negócio turbulento". À primeira vista, os valores ágeis podem parecer negligentes com questões válidas como atender prazos e criar um software financeiramente viável, entretanto, a agilidade está preocupada com prazo e custo tanto quanto as metodologias tradicionais, porém ela busca durante todo o desenvolvimento equilibrar essas questões.

A forma como os valores do manifesto ágil é descrita, exemplo: "Indivíduos e interações mais que processos e ferramentas", "o mais que" marca uma divisão de importância entre itens à direita do "mais que" e itens a esquerda do "mais que", a agilidade tem como filosofia entender a importância dos itens à direita, contudo valorizar mais os itens à esquerda. Oliveira (B. C. OLIVEIRA, 2018) destaca que o manifesto ágil não negligencia processos, documentação ferramentas e muito menos planejamento, mas revela que eles possuem uma importância secundária fortalecendo o conceito de que seja gerado e utilizado apenas o essencial.

Além dos valores, foram definidos doze princípios ágeis para materializar os valores e auxiliar na difusão das suas ideias:

- Nossa maior prioridade é satisfazer o cliente, através da entrega adiantada e contínua de software de valor;
- Aceitar mudanças de requisitos, mesmo no fim do desenvolvimento. Processos ágeis se adequam a mudanças, para que o cliente possa tirar vantagens competitivas;
- Entregar software funcionando com frequência, na escala de semanas até meses, com preferência aos períodos mais curtos;
- Pessoas relacionadas à negócios e desenvolvedores devem trabalhar em conjunto e diariamente, durante todo o curso do projeto;
- Construir projetos ao redor de indivíduos motivados. Dando a eles o ambiente e suporte necessário, e confiar que farão seu trabalho;

- O método mais eficiente e eficaz de transmitir informações para, e por dentro de um time de desenvolvimento, é através de uma conversa cara a cara;
- Software funcional é a medida primária de progresso;
- Processos ágeis promovem um ambiente sustentável. Os patrocinadores, desenvolvedores e usuários, devem ser capazes de manter indefinidamente, passos constantes;
- Contínua atenção à excelência técnica e bom design, aumenta a agilidade;
- Simplicidade: a arte de maximizar a quantidade de trabalho que não precisou ser feito;
- As melhores arquiteturas, requisitos e designs emergem de times auto-organizáveis;
- Em intervalos regulares, o time reflete em como ficar mais efetivo, então, se ajustam e otimizam seu comportamento de acordo.

Os ganhos obtidos ao se adotar uma metodologia baseada nesses princípios são numerosos como a melhoria na comunicação, inspeção e adaptação constantes em prol da melhoria contínua, aumento do comprometimento por parte da equipe, redução dos desperdícios, mitigação dos riscos, melhor reação diante de mudanças acompanhada de um redução de tempo para contornar desvios, aprendizado constante e maior rapidez na tomada de decisões (J. F.OLIVEIRA, 2014).

2.3.2 Características das Metodologias Ágeis

Através dos valores e princípios é possível compreender os comportamentos esperados em projetos que utilizam modelos ágeis. Entretanto, o manifesto ágil não define regras ou métodos pelos quais os projetos possam atender ao pensamento agilista. Sendo assim, metodologias e modelos ágeis foram surgindo a fim de personificar a filosofia em atividades práticas. Conforme estes princípios, modelos ágeis fundamentam-se em uma abordagem incremental para a especificação, desenvolvimento e a entrega do software (SOMMERVILLE, 2011).

Outra característica das metodologias ágeis é que elas são adaptativas em vez de prescritivas, por isso a melhoria contínua (busca constante de um estado melhor através de mudanças e transformações contínuas) é incentivada através de ciclos de inspeção e adaptação.

Por esse motivo, as metodologias ágeis utilizam processos empíricos em vez de prescritivos (LARMAN, 2003).

Por definição, o empirismo é o pensamento filosófico que defende que toda nossa estrutura de cognição é formada através de experiências, ou seja, nosso conhecimento é construído a partir do que vivenciamos, quanto mais vastas e intensas forem essas experiências maior é a profundidade do processo cognitivo. Traduzindo para o contexto de desenvolvimento de software, processos empíricos são aqueles em que o aprendizado sobre o sistema se dá através de lições aprendidas durante todo o processo de desenvolvimento.

Enquanto processos prescritivos são aconselhados para atividades que podem ser alcançadas através de uma sequência de passos ordenados, os processos empíricos são apropriados para ambientes instáveis e com um alto índice de mudanças (LARMAN, 2003).

A premissa nas metodologias ágeis é de que sempre haverá mudanças no projeto impossíveis de serem previstas em seu início, e essas mudanças são bem-vindas, estando todos os envolvidos preparados para elas. Diferente das metodologias tradicionais que focam no processo e acreditam que um único processo pode atender diferentes níveis de projetos e pessoas, os modelos ágeis afirmam que nenhum processo jamais poderá ser equiparado à habilidade da equipe de desenvolvimento. Logo, a função do processo é suportar a equipe de desenvolvimento e seu trabalho (BESSA & ARTHAUD, 2018).

2.4 **Encerramento do Capítulo**

O pensamento ágil abraça a complexidade, ao invés de evitá-la através de rotinas burocráticas, porém, a agilidade não define como os projetos podem colocar em prática valores e princípios de forma a obterem melhores resultados. Sendo assim, os modelos ágeis começaram a surgir com o objetivo de traduzir em “Como Fazer” a agilidade. No próximo capítulo, será apresentado a definição dos modelos ágeis *Scrum*, *Kanban* e *Lean*, os quais foram escolhidos devido seus resultados positivos e popularidade.

3 **MODELOS ÁGEIS**

Neste capítulo será apresentado o conceito de modelos ágeis, as motivações por trás da criação destes modelos e uma breve descrição dos modelos escolhidos para este trabalho. Em seguida, os modelos ágeis *Scrum*, *Lean* e *Kanban* serão detalhados com foco em compreender histórico, motivação, estrutura e a contribuição de suas aplicações para o mundo de desenvolvimento de software.

De acordo com Sommerville (SOMMERVILLE, 2011), às metodologias pesadas eram inadequadas para os projetos de software, pois o tempo gasto com as atividades de desenvolvimento era menor que o gasto com elaboração de documentação. A insatisfação gerada por essa desarmonia presente nas abordagens pesadas levou um grande número de profissionais de tecnologia da informação a proporem novas formas de trabalhar adotando a mentalidade ágil, originando assim modelos ágeis de trabalho. Estes novos modelos procuraram criar abordagens onde a equipe de desenvolvimento pudesse focar no software em si, e não em seu planejamento e documentação.

Os modelos ágeis que surgiram buscavam resolver cada um, problemas relacionados a um determinado ambiente ou projeto, Presner e Junior (PRESNER & JUNIOR, 2014) afirmam que apesar de existirem diversas abordagens, existem características que são fundamentais a todas elas:

- Paralelismo entre atividades: As atividades de especificação, projeto e implementação são executadas em paralelo, trazendo dinamismo para o ritmo do projeto e permitido maior sincronia e como consequência maior assertividade.
- Desenvolvimento incremental: As entregas do software são divididas em incrementos de forma que possam ser melhor inspecionadas pelas partes interessadas.
- Ferramentas interativas: Utilização de ferramentas que permitam que o projeto seja construído em colaboração com todas as partes interessadas.

Os modelos ágeis são um conjunto de práticas que materializam os valores e princípios ágeis. A forma como a agilidade é aplicada ao desenvolvimento de software será discutida através de três modelos bastantes populares, são eles: *Scrum*, *Kanban* e *Lean*.

Scrum

O *Scrum* é o modelo mais popular quando se trata de processos ágeis (RUBIN, 2012). Ele é fundamentado no empirismo e tem como premissa que o desenvolvimento de software é um processo complexo e imprevisível, portanto, o *Scrum* acredita que a aplicação de práticas empíricas de controle de processo são capazes de garantir visibilidade, inspeção e adaptação (MAHNIC & DRNOVSCEK, 2005).

Embora o *Scrum* tenha sido criado como um modelo de desenvolvimento de software, por ter uma estrutura simplificada e um conjunto pequeno de regras, seu uso pode ser feito em diferentes tipos de organizações, tanto públicas quanto privadas, instituições de ensino, saúde e os mais diversos ambientes de negócio, porém sempre com cautela para promover eventuais adaptações que sejam necessárias (SIMOYAMA et al., 2016)

Para tratar questões mais específicas do desenvolvimento de software, como: prazo, requisitos, tecnologias e ferramentas. O *Scrum* adota uma postura de inspeção constante, com o objetivo de se adaptar às mudanças com flexibilidade (MAHNIC & DRNOVSCEK, 2005).

Kanban

Apesar da grande popularidade do *Scrum*, outros modelos vêm ganhando espaço, como é o caso do *Kanban* (SANTOS, 2015). O *Kanban* surgiu para atender uma demanda da indústria para controle de fluxo de material, ele é uma das ferramentas utilizadas pelo sistema *Just In Time* (JIT), explicado mais à frente, que por sua vez é um dos alicerces do Sistema Toyota de Produção (STP) (ALMEIDA, 2015).

O *Kanban* é um sistema de comunicação e gestão através de cartões. Segundo Barros e Junior (BARROS & JUNIOR, 2011), o *Kanban* tem por principais características limitar o nível máximo de estoque, controlar e puxar a produção. Embora o *Kanban* tenha sido criado em um contexto industrial, devido ao seu sucesso, ele foi adaptado para outros domínios, inclusive para a TI, tornando-se um eficiente sistema para gestão de desenvolvimento de software.

A aplicação do *Kanban* para o desenvolvimento de software possui algumas diferenças em relação a utilizada na indústria. Sendo assim, o *Kanban* foi adaptado para o contexto de desenvolvimento de sistemas, pela primeira vez, por David J. Anderson, neste novo cenário, o *Kanban* não possui muitas restrições e tem a capacidade de adaptar-se à realidade de cada organização. Porém, ainda na visão de Anderson, para que a implantação do *Kanban* seja bem-sucedida é preciso que exista um processo pré-estabelecido (GIRARDI, 2016).

Lean

O *Lean* Manufacturing (ou Produção Enxuta ou Manufatura Enxuta) é uma filosofia de gestão inspirada em práticas e resultados desenvolvidas pelo Sistema Toyota de Produção com a finalidade de melhorar o sistema de produção industrial através da eliminação de desperdícios, aumento da qualidade e diminuição de custos (NETTO, 2017). A atual popularidade do *Lean* está diretamente ligada a seus resultados positivos. Esta filosofia foi ganhando espaço em organizações no mundo inteiro ao longo dos últimos anos e atualmente é utilizada por diversas indústrias manufatureiras as quais se beneficiam das oportunidades de melhorias oferecidas pelo sistema (SAIA, 2009).

Diante dos resultados expressivos na manufatura, Womack e Jones, demonstraram grande interesse sobre o potencial da filosofia *Lean* e em 1996, lançaram o livro "*Lean Thinking: Banish Waste and Create Wealth in Your Corporation*" (WOMACK & JONES, 1996) que expunha um pensamento que o *Lean* poderia extrapolar a manufatura e ser aplicado em todos os tipos de indústrias, de produção de produto ou serviço e que aquilo que demonstrava ser tão bem sucedido à produção em massa também poderia evoluir para se adaptar a outras realidades (Dantas,2016).

O crescimento da TI nas empresas junto com o aumento do seu valor estratégico, também vem acompanhado com o crescimento de um custo necessário para sustentar uma tecnologia de informação adequada. Neste ponto, o *Lean* pode ser um aliado para que as empresas prosperem aplicando uma mentalidade que evita desperdícios e reduz custos.

3.1 ***Scrum***

Scrum é um modelo para desenvolver, entregar e manter produtos complexos (SCHWABER & SUTHERLAND, 2017). Ele tem sido usado para gerenciar milhares de projetos com sucesso há mais de 10 anos em centenas de organizações. O *Scrum* foi concebido para funcionar em cenários que é impossível prever o que irá acontecer em seguida, por isso foi estruturado de forma a oferecer um conjunto de práticas que mantêm o processo visível aos envolvidos (SCHWABER, 2004).

O *Scrum* é utilizado em sua maioria em cenários de desenvolvimento de software, apesar de ser adaptável a outros ambientes. Ao se deparar com a popularidade do *Scrum* e seu largo uso em organizações do mundo inteiro, pode-se pensar que seu sucesso se deve a capacidade de garantir que o projeto sairá exatamente de acordo com os resultados previstos, contudo, ao lançar um olhar mais profundo sobre sua aplicação é mais correto pensar que ele orienta o progresso em direção ao resultado mais valioso possível para parte interessadas (SCHWABER, 2004).

3.1.1 Histórico

O termo *Scrum* foi popularizado por Jeff Sutherland e Ken Schwaber na década de 90, a palavra "*scrum*" é o nome dado a uma formação para reinício de jogada no rúgbi. A expressão foi adotada para batizar o modelo, após ser usada no artigo *The New New Product Development Game* (TAKEUCHI & NONAKA, 1986), em tradução O Novo Jogo no Desenvolvimento de Novos Produtos, publicado na revista *Harvard Business Review* por Takeushi e Nonaka em 1986. Nesse artigo, os autores utilizaram uma analogia com o rúgbi para ilustrar a forma como a criação de software estava sendo conduzida pelos times de desenvolvimento na década de 80.

Entretanto, há alguns indícios que foi no livro *Wicked problems, righteous solutions* (DeGRACE & STAHL, 1990), em tradução Problemas perversos, soluções corretas, de 1990, que aparece pela primeira vez a ideia de aplicar no desenvolvimento de software o conjunto de práticas descritas por Takeushi e Nonaka. Os autores DeGrace e Stahl, batizaram essa nova forma de trabalhar de "*Scrum*", em resumo, o livro relata a ineficiência do modelo cascata para gerenciar a construção de um software e oferece possíveis alternativas, entre elas o *Scrum* (SABBAGH, 2018).

> Trecho do livro *Wicked Problems, Righteous Solutions*: Se *Scrum* fosse aplicado ao desenvolvimento de software, seria mais ou menos assim: (...) forme a equipe escolhendo cuidadosamente uma pessoa de cada uma das [fases tradicionais de desenvolvimento]. (...) Você então dá a eles uma descrição do problema a ser resolvido e (...) os desafia, dizendo que deverão produzir o sistema em, digamos, metade do tempo e dinheiro e que deve ter o dobro da performance de outros sistemas. Em seguida, você lhes diz que como eles farão isso é da conta deles (DeGRACE & STAHL, 1990).

No livro de DeGrace e Stahl há apenas concepção que software é um tipo de problema que não pode ser obviamente definido e cada mudança no desenvolvimento altera a definição

do problema a ser resolvido, por isso o termo “problema perverso" para expressar a complexidade de conceber um software. Apesar das divergências quanto à origem do termo, não há dúvidas que coube a Sutherland e Schwaber o esforço de transformar o conceito em um processo funcional para lidar com as complexidades da construção de sistemas (SABBAGH, 2018).

3.1.2 Estrutura

A definição e práticas do *Scrum* são descritas no documento oficial do processo, o Guia *Scrum* (SCHWABER & SUTHERLAND, 2017), este documento é mantido e atualizado até hoje por seus criadores. O *Scrum* consiste em papéis, eventos, artefatos e regras associados entre si. Cada elemento serve a um propósito específico e é fundamental para a aplicação e êxito do modelo de trabalho. O *Scrum* adota uma abordagem iterativa e incremental como estratégia para diminuir incertezas e controlar riscos, além disso, é fundamentado no empirismo, possuindo forte cultura de feedback.

Em resumo, a estrutura do *Scrum* (figura 2) funciona da seguinte forma: no início de uma iteração, o time analisa a lista de coisas que precisam ser feitas, logo após, ele seleciona o que acredita ser possível de ser entregue em formato de um incremento funcional ao final de uma interação. Após a decisão do que será feito, o time trabalha o restante da interação e no fim do ciclo, o trabalho realizado é apresentado às partes interessadas para ser inspecionado e receber comentários sobre possíveis adaptações (SCHWABER, 2004).

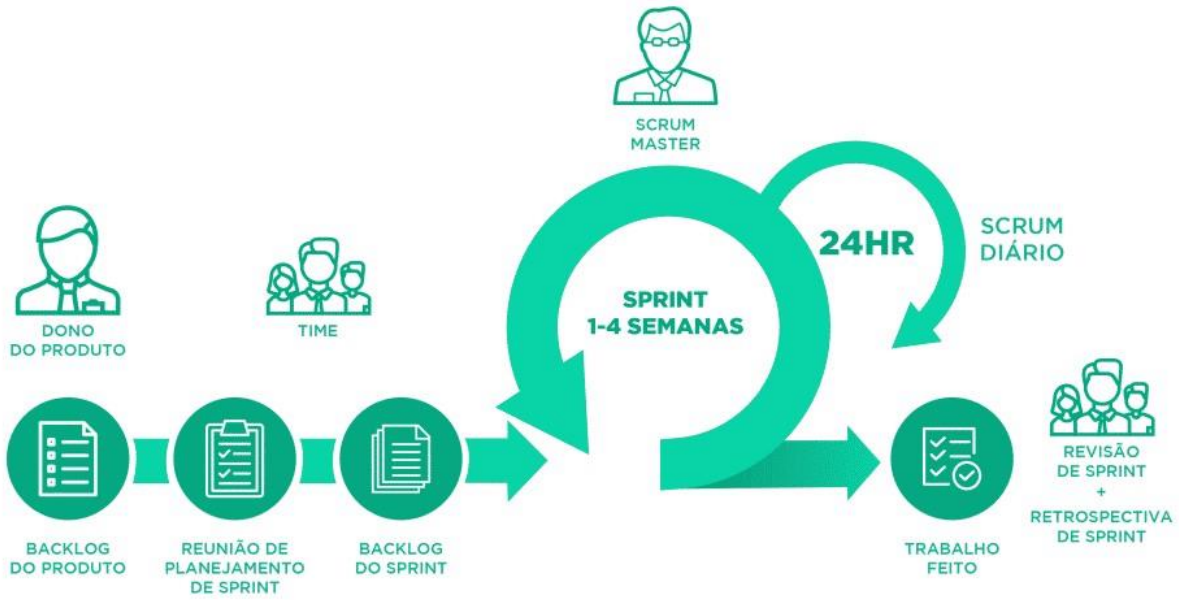

Figura 2 - Processo *Scrum*. (Fonte: BARBOSA, 2020)

3.1.2.1 Papéis

A base do *Scrum* é um pequeno time de pessoas, esse time é altamente flexível, adaptativo e composto por três papéis: *Product Owner*, *Scrum Master* e Time de Desenvolvimento. O *Product Owner*, em tradução Dono do Produto, é o indivíduo encarregado por maximizar o Retorno de sobre Investimento (ROI) do produto, ele é o único responsável por gerenciar o *Backlog* do Produto e deve sempre estar atento no que o produto precisar ter ou não para atender as exigências do cliente e/ou mercado. O *Scrum* Master, não há uma tradução exata para esse papel, desempenha a função de garantir que o *Scrum* seja entendido e seguido pelos indivíduos envolvidos no processo; já o Time de Desenvolvimento, consiste em profissionais que executam o trabalho necessário para entregar um incremento do produto ao final de cada *Sprint*.

O *Scrum* transforma pequenas equipes em gerentes de seu próprio destino (SCHWABER, 2004). A equipe do *Scrum* é auto gerenciável, isso significa que o próprio time se organiza e planeja para alcançar os objetivos. Um time deve possuir apenas um *Scrum Master* e um Dono do Produto, entretanto, o Time de Desenvolvimento pode ser composto por três a nove integrantes. Times pequenos demais ou grandes demais podem possuir problemas de

comunicação, tomada de decisão e gerar muita complexidade para um processo empírico, por isso o *Scrum* fixa limites de integrantes, o objetivo é ter um tamanho que consiga ser ágil o bastante mas possua todas as habilidades para concluir o trabalho a ser entregue (SCHWABER & SUTHERLAND, 2017).

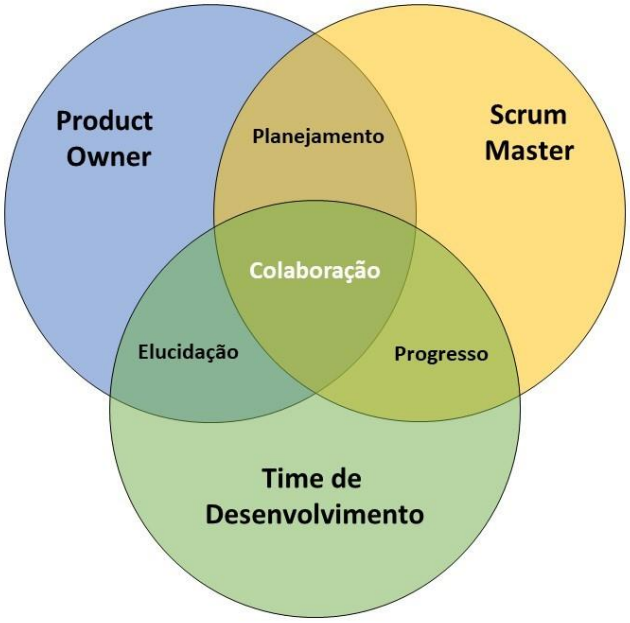

Figura 3 - Processo de Comunicação Colaboração no *Scrum*. (Fonte: Adaptado do KNOWLEDGE 21, 2020)

Os papéis colaboram entre si de forma que as interações entre eles fomentam ações essenciais para o resultado do projeto, exemplificado na figura 3. As atividades que o *Scrum Master* e Dono do Produto realizam durante o projeto contribuem para o planejamento do mesmo, exemplo de atividades são: manter o modelo de trabalho funcionando, priorizar a lista de requisitos do sistema e colaborar com as partes interessadas. Já as interações entre Dono do Produto e Time de Desenvolvimento são importantes, para que todos estejam alinhados quanto ao que precisa ser construído e o valor agregado ao final de cada interação. Por fim, as interações do *Scrum Master* com o Time de Desenvolvimento são em sua maioria ligadas ao êxito do planejamento, visando garantir que os eventos estão cumprindo com seus propósitos de colaboração, inspeção e adaptação de acordo com as definições do *Scrum*.

3.1.2.2 Eventos

Todos os eventos do *Scrum* são eventos *time-boxed*, ou seja, possuem uma duração máxima preestabelecida. A existência dos mesmos tem como o objetivo garantir uma regularidade e minimizar a necessidade de reuniões extras ao processo do *Scrum* (SCHWABER & SUTHERLAND, 2017). A *Sprint* é o coração do *Scrum*, ela é o contêiner que engloba o processo, é durante a *Sprint* que ocorrem os demais eventos e elaboração dos artefatos. Uma *Sprint* pode durar de 1 a 4 semanas, contudo deve manter uma duração consistente durante todo o projeto. A execução de um projeto pode exigir uma quantidade indeterminada de *Sprint*s e elas podem e devem acontecer quantas vezes forem necessárias para a conclusão.

Compondo a *Sprint*, existem os eventos de: Planejamento da *Sprint*, Reunião Diária, Revisão da *Sprint* e Retrospectiva da *Sprint*. O primeiro evento de uma *Sprint* é Planejamento da *Sprint*, onde o trabalho a ser realizado durante a *Sprint* pelo Time de Desenvolvimento é planejado, ao menos parcialmente. Durante o evento, o plano de trabalho gerado durante essa reunião é criado de forma colaborativa por todo o time *Scrum*. Durante o planejamento também é acordado qual a Meta da *Sprint*, sendo este objetivo o que o time deve alcançar ao final da iteração. O *time boxed* para este evento é de no máximo 8 horas para Sprint*s* de um mês, para *Sprints* menores o tempo é proporcionalmente menor.

A Reunião Diária, como o nome já diz acontece todo o dia e permite que o time planeje o trabalho para as próximas 24 horas, esta reunião tem como objetivo otimizar a colaboração e examina o progresso do trabalho para garantir que o time está na direção que o permita atingir a Meta da *Sprint*. As reuniões diárias são curtas, possuindo um *time-boxed* de no máximo 15 minutos.

A Revisão da *Sprint* é o evento em que o time *Scrum* e partes interessadas inspecionam o incremento do produto resultado da *Sprint* e colaboram sobre o que foi feito e o que fazer a seguir para garantir o sucesso do projeto, este evento ocorre ao final de cada *Sprint*. Para os eventos de Revisão da *Sprint* o *time-boxed* é de no máximo 4 horas para *Sprints* de um mês, para Sprint*s* menores o tempo é proporcionalmente menor.

Também ao final da *Sprint* acontece a Retrospectiva da *Sprint*, neste evento o time *Scrum* se reúne para realizar uma auto inspeção, neste momento há uma reflexão sobre aspectos positivos e negativos que ocorreram durante a *Sprint* e a elaboração de um plano de melhorias para a próxima iteração. A retrospectiva é o principal evento onde o *Scrum* aplica o conceito

de melhoria contínua, para que o evento funcione adequadamente é papel do *Scrum Master* garantir um ambiente positivo onde todos se sintam à vontade para se expressar. Para este evento, o *time-boxed* é de no máximo 3 horas para Sprints de um mês, para Sprints menores o tempo é proporcionalmente menor.

3.1.2.3 Artefatos

Os artefatos representam o trabalho a ser executado ou o resultado produzido, o *Scrum* possui três artefatos: *Backlog* do Produto, *Backlog* da *Sprint* e o Incremento (ou Trabalho Feito). O *Backlog* do Produto é uma lista ordenada com tudo que é necessário que o resultado do projeto contenha, a lista reúne todas as características, funções, requisitos, melhorias e correções que devem ser feitas no produto. Durante a execução do projeto novas necessidades podem ser descobertas e adicionadas ao *Backlog* do Produto, portanto esta é uma lista viva que se adapta às demandas.

O *Backlog* da *Sprint* é um conjunto de itens selecionados para serem executados durante a *Sprint* corrente, esses itens são oriundos do *Backlog* do Produto, durante o Planejamento da *Sprint* os itens selecionados são refinados de forma que o Time de Desenvolvimento consiga compreender o trabalho futuro. Ainda assim, durante a *Sprint* podem ser feitas novas descobertas, elas são acrescida ao *Backlog* da *Sprint*. Assim como o *Backlog* do Produto, o *Backlog* da *Sprint* também é vivo e permite que o time lide com as mudanças com maior plasticidade.

Por último, temos o Incremento ou Trabalho Feito, ele é resultado da soma de todos os itens do *Backlog* do Produto completados durante a *Sprint* e devidamente integrados aos incrementos de todas as Sprints anteriores. O Incremento é o artefato que é inspecionado durante a Revisão da *Sprint*, ele é um passo na direção do objetivo do projeto. Para o *Scrum* o Incremento precisa ser funcional e potencialmente liberável, isso significa que, caso seja liberado para o mercado, os usuários conseguirão utilizá-lo de forma completa.

3.1.2.4 Regras

O *Scrum* não possui muitas regras, contudo as existentes precisam ser seguidas para que o processo funcione. O *Scrum Master* é o responsável por zelar pela correta implementação do modelo, qualquer indicativo que o processo não está sendo seguido com definido pelo Guia *Scrum*, o *Scrum Master* deve ensinar e mostrar o propósito de cada coisa. Além dos papéis, eventos e artefatos, o *Scrum* também realiza apontamentos relevantes sobre os seguintes elementos: *time-boxed*, transparência do artefato, definição de pronto e responsabilidades e ferramentas.

Time-boxed como já exposto neste trabalho, é o período máximo em que um determinado evento pode ocorrer. Cabe ao *Scrum Master* garantir que o *time-boxed* seja respeitado e guiar as equipes para se manterem focadas nas finalidades de cada evento. Acerca da Transparência do Artefato, o *Scrum* defende que quando os artefatos estão visíveis e são compreendidos por todos os envolvidos é possível tomar decisões sólidas. Quando não existe visibilidade, a incerteza impacta diretamente a qualidade das decisões colocando o sucesso do projeto em risco.

A Definição de Pronto é um acordo que o time *Scrum* realiza entre si para identificar que uma atividade está completa. A Definição de Pronto pode ser entendida como uma lista de verificação de atributos que determinada atividade deve atender para ser entregue, é através da Definição de Pronto que o time pode garantir a qualidade da entrega de suas tarefas. Conforme o time ganha maturidade, essa lista pode ser alterada para atender cada vez mais critérios rigorosos de qualidade.

Cada papel no *Scrum* possui responsabilidades, os papéis podem e devem colaborar entre si e até mesmo assumir determinadas atividades de um outro papel. Porém, a responsabilidade não é delegada, por mais que uma atividade esteja sendo realizada por outra pessoa ou papel, o responsável original pela atividade ainda continua respondendo pela conclusão e qualidade da mesma.

Por fim, no quesito de ferramentas o Guia *Scrum* não define qualquer tipo de ferramenta, para auxiliar na execução de atividades. O time é livre para escolher quaisquer recursos que os ajudem durante o andamento do projeto. O que se espera das ferramentas escolhidas é que elas sejam capazes de suportar o processo e não modificá-lo.

3.1.3 Contribuição

O *Scrum* se tornou tão popular devido a sua abordagem para lidar com as mudanças. Ele tem como pilares: transparência, inspeção e adaptação. Para o *Scrum*, a transparência é a qualidade de manter todos os processos visíveis a todos os responsáveis, a inspeção são as verificações para detectar variações indesejadas e a adaptação é a capacidade de ajuste para minimizar os desvios em tempo de desenvolvimento (SCHWABER & SUTHERLAND, 2017).

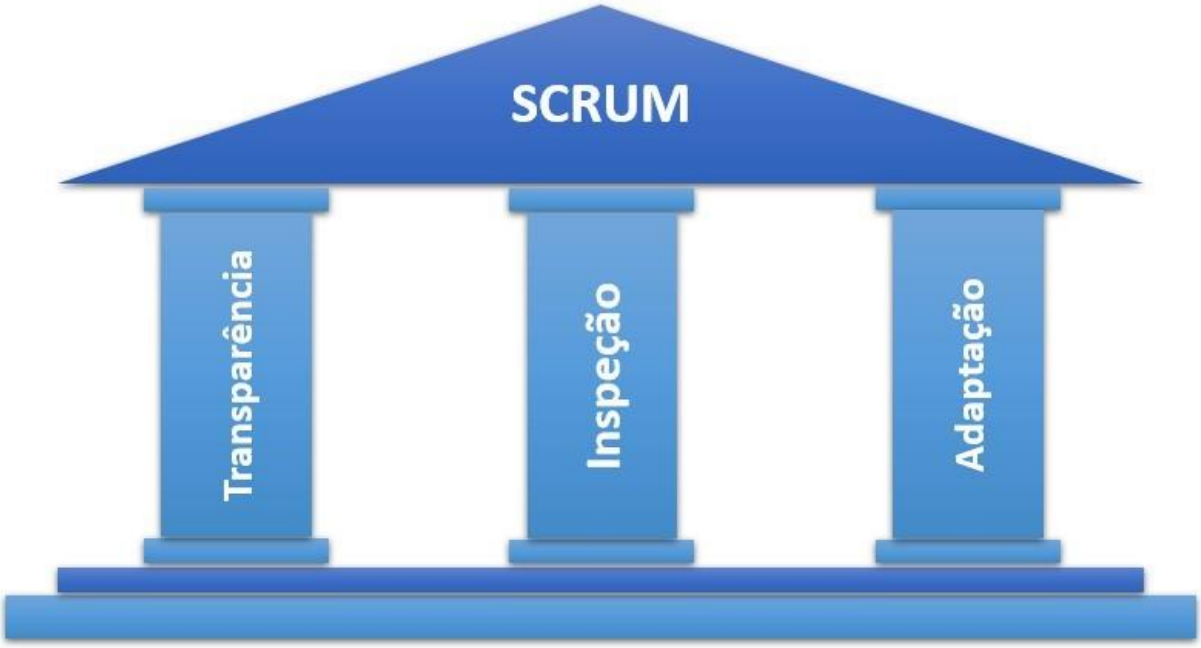

Figura 4 - Pilares do *Scrum*. (Fonte: Adaptado do blog KNOWLEDGE 21, 2019)

Os três pilares (figura 4) podem ser experienciados em todos os eventos, artefatos e papéis do *Scrum*. Os artefatos do *Scrum* são itens vivos que estão abertos constantemente à inspeção tanto nos eventos preestabelecidos no processo quanto em oportunidades que podem vir a acontecer durante a interação. Os eventos do *Scrum* acontecem de forma regular e com um propósito pré-definido dando visibilidade do que é esperado aos participantes. O *Scrum* incentiva a colaboração e *feedback* constante, visando a inspeção e adaptação, tanto dentro do time *Scrum* quanto com as partes interessadas e, além disso, o papel do *Scrum Master* deve garantir que o processo e seus resultados sejam transparentes para os envolvidos.

Uma característica bem-sucedida no *Scrum* é como ele mitiga o risco. A duração de uma *Sprint* é de no máximo quatro semanas, ela é projetada dessa forma para minimizar os riscos que

um planejamento a longo prazo oferece, como baixo entendimento de escopo e mudanças no negócio. Quando se está lidando com uma economia altamente transformadora e com tecnologias que evoluem a todo momento, progredir através de ciclos curtos de aprendizado é uma abordagem assertiva de solução de problemas testada e comprovada com êxito pelo *Scrum* (SCHWABER, 2004).

Além dos ciclos de entrega constante, o *Scrum* não se concentra em entregar qualquer incremento, ele busca entregar o incremento de valor com mais alta prioridade de acordo com a definição do Dono do Produto. E isto só pode ser feito quando há intensa colaboração entre as partes, por isso, um dos fatores que faz com que o *Scrum* funcione é que ele reúne a inteligência de muitas mentes sobre um problema, criando um ambiente onde todos os envolvidos têm voz e podem examinar as mais diversas saídas para os problemas encontrados. Quanto mais complexos são os problemas, maior é a necessidade da distribuição de responsabilidades para os agentes que estão próximos do problema (SCHWABER, 2004).

O *Scrum* está altamente acoplado aos valores e princípios ágeis. A estrutura do *Scrum* é focada em auxiliar os envolvidos a tomar decisões baseadas em fatos por meio de um processo contínuo de aprendizado, além disso a cultura de colaboração no desenvolvimento, a autonomia dada ao time e a busca da satisfação do cliente fundamentam o processo. Quando todos esses fatores funcionam juntos é difícil o *Scrum* não ter sucesso (SCHWABER, 2004).

3.2 ***Kanban***

A eficácia do *Kanban* se dá através do baixo investimento acompanhado por melhorias na produtividade, eliminação de desperdícios alinhadas com uma maior colaboração entre os indivíduos envolvidos (CADORIN, 2015). O *Kanban* disponibiliza uma série de vantagens sobre as formas de controle de produção tradicionais, ele possui uma simplicidade que permite lançar luz sobre o processo produtivo a fim de compreender melhor as falhas e problemas existentes (PEREIRA, 2018).

3.2.1 Histórico

O *Kanban* nasceu para resolver uma necessidade complexa originada nas linhas de produção da Toyota na década de 60. Ele é um método que faz parte da metodologia *Just In Time* e foi criado com a finalidade de controlar os fluxos de produção. Para funcionar, a metodologia JIT se utiliza de diversos métodos, sendo um deles o *Kanban*. Contudo, para ser possível compreender os objetivos na aplicação do *Kanban*, antes é preciso entender as motivações do sistema JIT.

3.2.1.1 *Just In Time*

Figura 5 - Representação de uma linha de produção no modelo fordista. (Fonte: Cena do filme Tempos Modernos, CHAPLIN, 1936)

Antes da criação do *Just In Time*, as linhas de produção industriais seguiam o modelo fordista de produção. O sistema fordista, em parte representado na figura 5, era extremamente ágil na produção dos produtos, nele a linha de produção era dividida em postos, em cada posto, os operários possuíam um tempo definido para realizar sua função, ao final deste tempo o produto seguia para a próxima parada, ao final da linha de produção, o produto era armazenado em estoque (figura 6) enquanto aguardava escoamento.

Figura 6 - Estoques fordistas. (Fonte: Autor desconhecido)

A japonesa Toyota que buscava fazer frente a indústria americana, percebeu que não era capaz de produzir em larga escala como os americanos devido à crise que o Japão atravessava, para neutralizar esse descompasso foi criado o *Just In Time*, em tradução literal "Na Hora Certa". O intuito do JIT é entregar o produto certo na hora certa evitando o surgimento de grandes estoques como no fordismo e permitindo que a Toyota reduzisse custos. De acordo com Richard (LUBBEN, 1989) o sistema JIT pode ser definido como:

> A filosofia da manufatura *Just In Time* é operar um sistema de manufatura simples e eficiente capaz de otimizar o uso dos recursos de capital, equipamento e mão de obra. Isto resulta em um sistema de produção capaz de atender as exigências de qualidade e de entrega de um cliente, ao menor custo[...] a meta do *Just In Time*[...], é eliminar qualquer função desnecessária no sistema de manufatura que traga os custos indiretos, que não acrescente valor a empresa, e que impeça melhor produtividade ou agregue despesas desnecessárias no sistema operacional de cliente. (LUBBEN, 1989)

O JIT é um sistema que apoia a cultura de eliminação de desperdícios, por desperdício pode-se entender como tudo que é feito durante o processo e não acrescenta valor ao produto (JUNIOR, 2003), o conceito de desperdício é aprofundado adiante no tópico 3.3.1.1. Em uma linha de produção que aplica o JIT, o produto não avança na produção sem que o anterior não tenha avançado, ou seja, dessa maneira um produto só consegue entrar na linha de produção caso um produto tenha sido consumido pelo mercado. Esse tipo de comportamento de produção é denominado produção puxada.

Quando se trata de ritmo de produção há dois conceitos predominantes: a Produção Puxada e a Produção Empurrada. A Produção Empurrada é a mais tradicional, neste tipo de produção o produtor busca antecipar as demandas do mercado, este é o sistema aplicado pelo fordismo. Para exemplificar, a indústria da moda utiliza a produção empurrada, ela produz novas coleções várias vezes ao ano, na esperança de que o mercado absorva. Há uma ação de "empurrar" o produto até as prateleiras na expectativa que o cliente o compre movido pelo desejo do consumo. Neste modelo, há a formação de estoques, contudo com esse sistema em geral há superprodução, pois, a demanda antecipada pode não se concretizar, gerando custos com estoque e desperdício de mercadorias.

Já a Produção Puxada é um conceito da filosofia *Lean* (ver tópico 3.3) que não utiliza estoques ou possui estoques bastante reduzidos, o cliente que aciona o gatilho para o início da produção de um novo produto. Quando um produto é adquirido pelo cliente, como por exemplo, um computador em uma prateleira de loja, o produto comprado abre espaço na prateleira e "puxa" um novo computador que estava no estoque para que ocupe lugar na prateleira, em consequência há um novo espaço vazio no estoque que será ocupado por um produto que estava na fábrica e assim sucessivamente. Esse movimento que o consumidor realiza de "puxar" o produto inicia a movimentação da cadeia de produção até o início da mesma.

Para que a Produção Puxada seja bem-sucedida, ela deve ser coordenada através de um sistema que alerte a produção que é necessário repor o produto. Porém, como gerenciar a comunicação e fluxo de produção de uma grande indústria automobilística como a Toyota? A solução foi encontrada através da criação do *Kanban*.

3.2.2 Estrutura

O *Kanban* nasceu a partir da ideia de funcionamento de um supermercado. As mercadorias compradas são registradas pelo caixa através de um cartão que indica a quantidade e a especificação da mercadoria, após esse registro, o cartão é levado ao departamento de compras que realiza a reposição. Para que o *Kanban* funcione é necessário que a produção esteja preparada para produzir as quantidades demandadas no momento necessário (MORALES & COIMBRA, 2017).

A palavra *Kanban* tem origem japonesa e significa “cartão”, esse modelo utiliza cartões aderentes a uma notação preestabelecida para sinalizar a necessidade de uma ação e sua prioridade. O processo é composto pelos seguintes elementos: Cartão, Colunas e Quadro (figura 7).

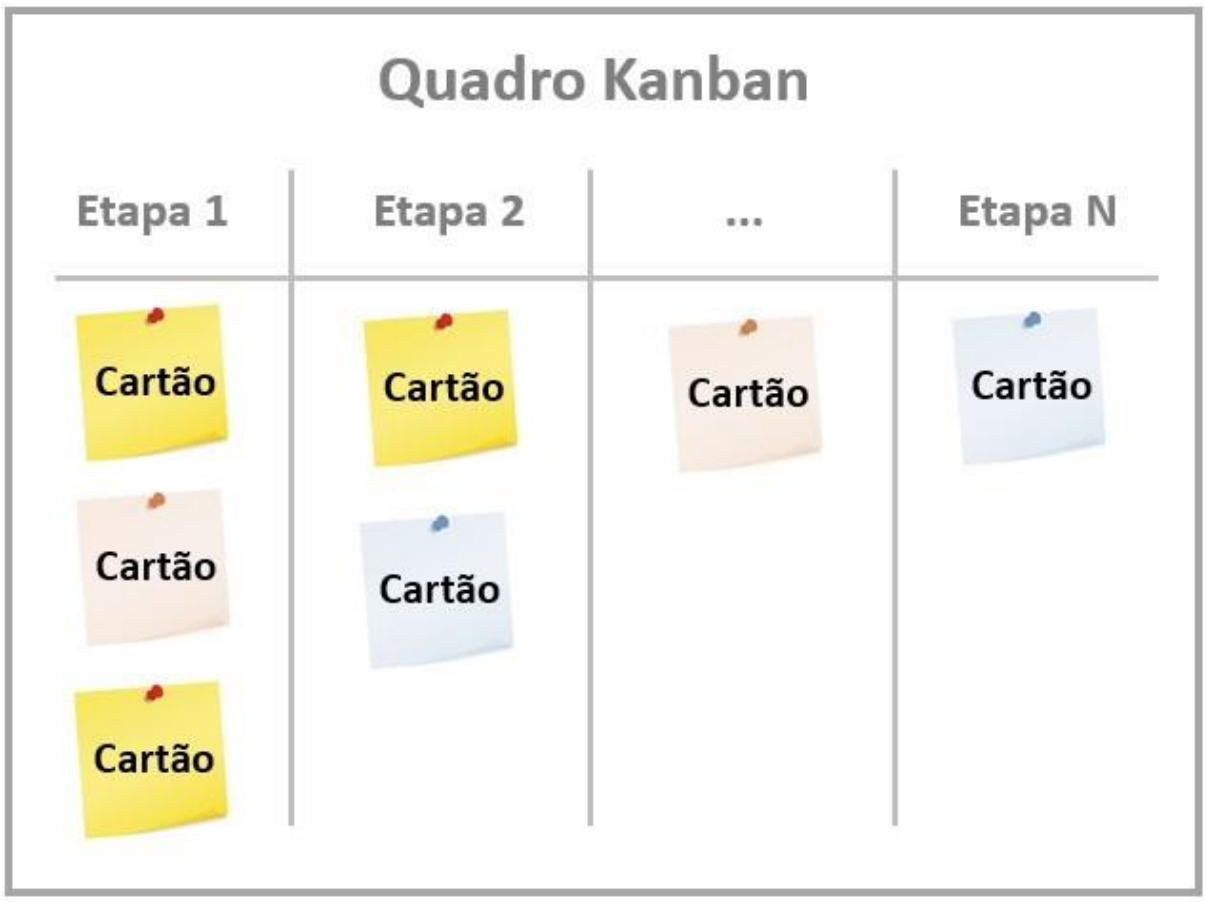

Figura 7 - Representação de um quadro *Kanban*. (Fonte: Adaptado do modelo *Kanban* do Sistema Toyota de Produção)

O Cartão é a menor parte do *Kanban*, ele representa uma ação ou tarefa que precisa ser executada para que um determinado resultado seja alcançado. Os cartões, seguem um esquema de notação escolhidos pela empresa, em geral, são adotados cores e/ou tamanhos que indicam sua prioridade de execução, responsável, tipo, ou qualquer outra informação estabelecida pela empresa. As Colunas indicam o fluxo de trabalho que os cartões devem seguir, as colunas podem ser criadas de acordo com a realidade da instituição. Os cartões são movidos pelas colunas, a fim de, indicar em qual etapa do processo estão, esse movimento permite a visualização do panorama do progresso da produção.

O Quadro é o elemento que personifica o *Kanban*, os cartões e colunas são fixados no quadro e cada quadro é um *Kanban*. Uma única equipe pode trabalhar com inúmeros quadros

Kanban simultaneamente. No sistema original, o quadro é de fato um artefato físico, contudo conforme o *Kanban* foi adotado por outros segmentos, os quadros foram ganhando versões digitais, este tipo de quadro virtual é chamado de *E-Kanban*, e é bastante utilizado no contexto de desenvolvimento de software.

Existem três tipos principais de *Kanban*: o de transporte ou movimentação, o de produção e o de fornecedor. O *Kanban* de transporte tem como finalidade avisar a etapa anterior que um estágio precisa de uma quantidade específica de determinado material. O *Kanban* de fornecedor serve para avisar aos fornecedores que certos insumos são necessários em determinados pontos da produção, e por último, o *Kanban* de produção gerencia as atividades que precisam ser realizadas para atender determinada demanda (VELOSO, 2006).

O *Kanban* do tipo de movimentação (figura 8) é o quadro da indústria idealizado originalmente pela Toyota, seu objetivo é o controle das entradas e saídas do estoque, a fim de equilibrar produção com o volume de produtos disponíveis para o mercado. Neste modelo, cada cartão contém a informação que indica a demanda de produção de um determinado produto. O *Kanban* de fornecedor é muito similar ao *Kanban* de movimentação, a diferença é que neste caso, funciona como um meio de comunicação externa, com agentes de fora da organização.

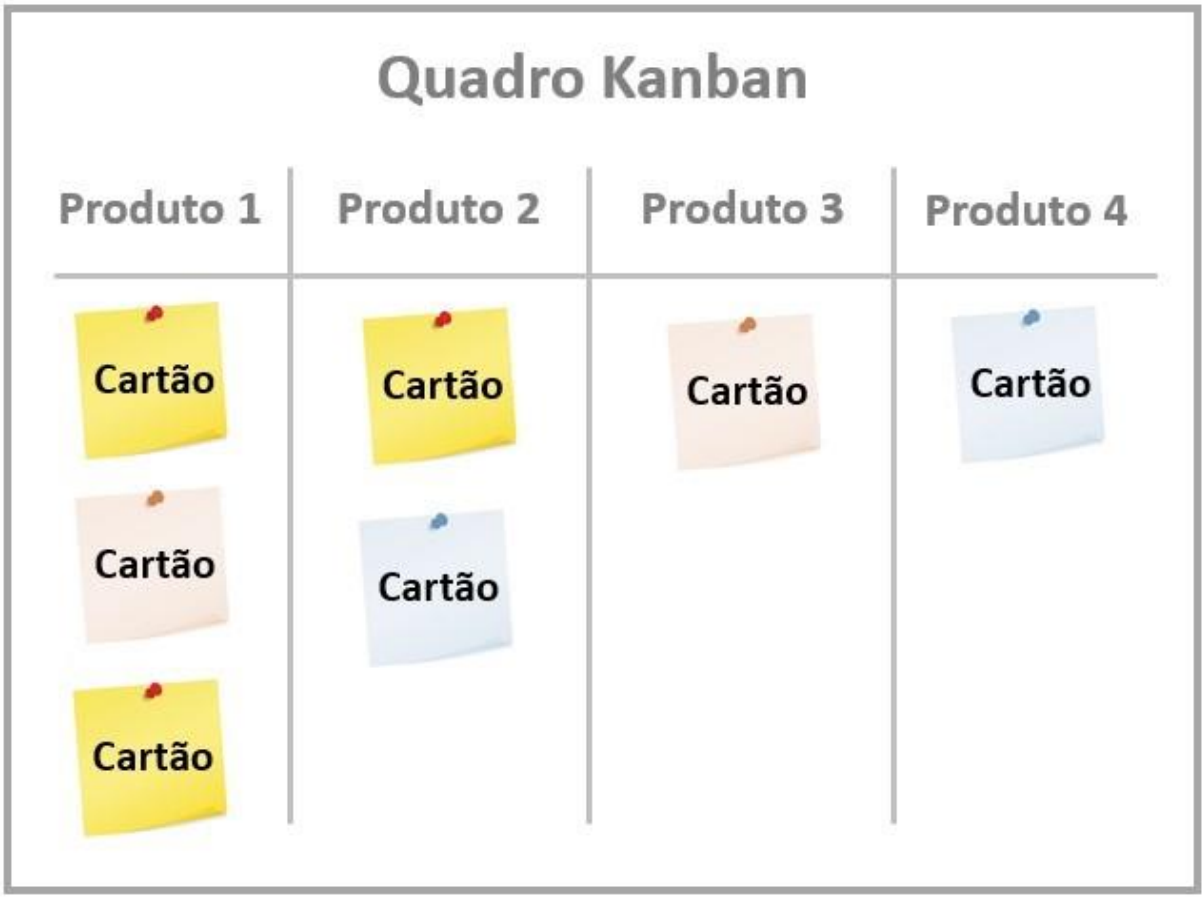

Figura 8 - Representação de um *Kanban* de movimentação. (Fonte: Adaptado do modelo *Kanban* do Sistema Toyota de Produção)

No *Kanban* de produção (figura 9) o foco é realizar a gestão de tarefas, nesta proposta existem três colunas “A Fazer”, “Em Execução” e “Feito”, no entanto nada impede que outras colunas sejam adicionadas. Cada coluna possui um conjunto de cartões que representam as atividades que precisam ser executadas e o time vai “puxando” as atividades de acordo com o fluxo de trabalho do negócio, por essa razão é denominado produção, pois cada atividade concluída é uma entrega.

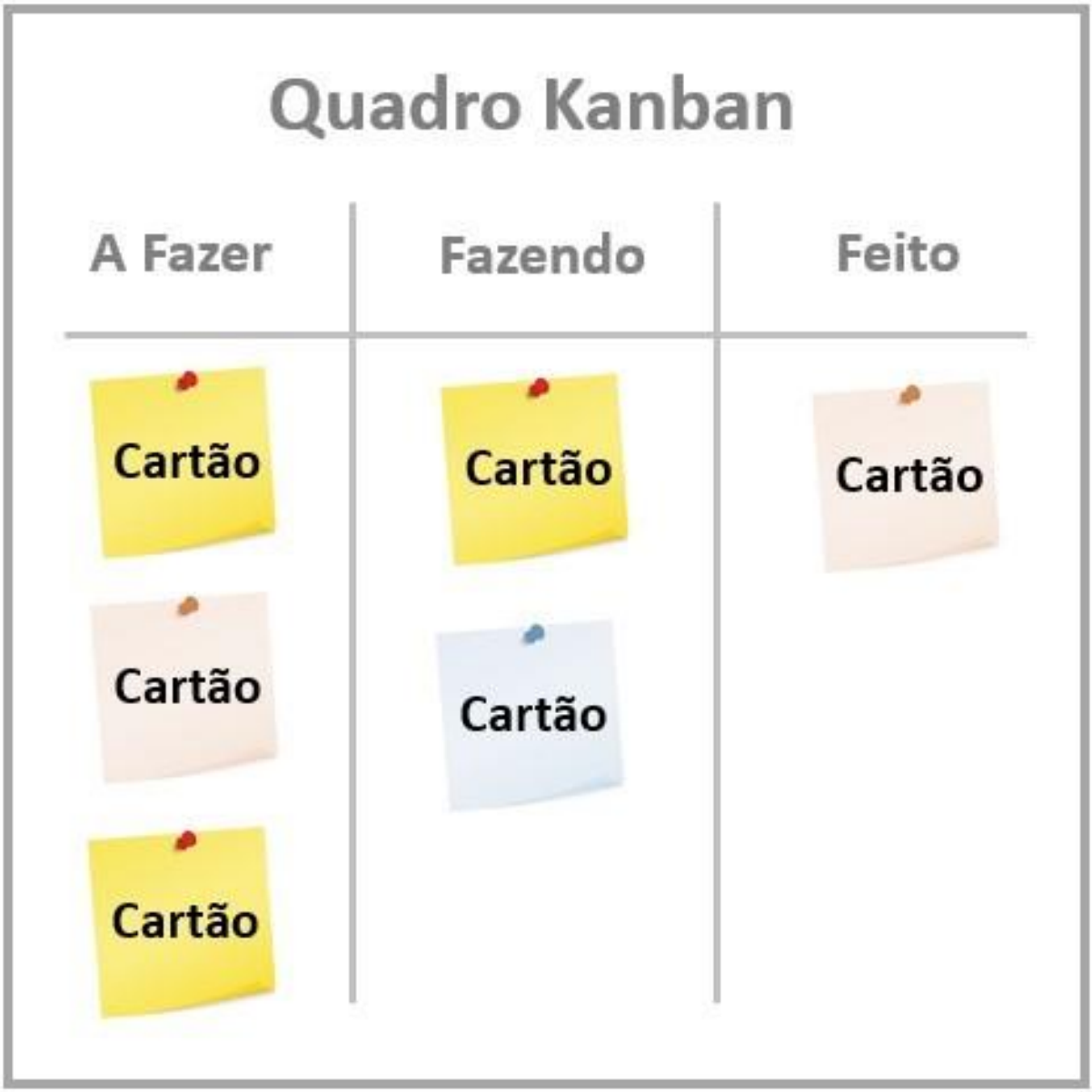

Figura 9 - Representação de *Kanban* de produção. (Fonte: Adaptado do modelo *Kanban* do Sistema Toyota de Produção

Este tipo de *Kanban* é o mais utilizado no contexto de desenvolvimento de software e empresas de prestação de serviço em geral. Entretanto, no desenvolvimento de sistemas ele sofre adaptações para refletir as etapas do processo de desenvolvimento de software praticadas pela a organização, um exemplo de adaptação representado na figura 10.

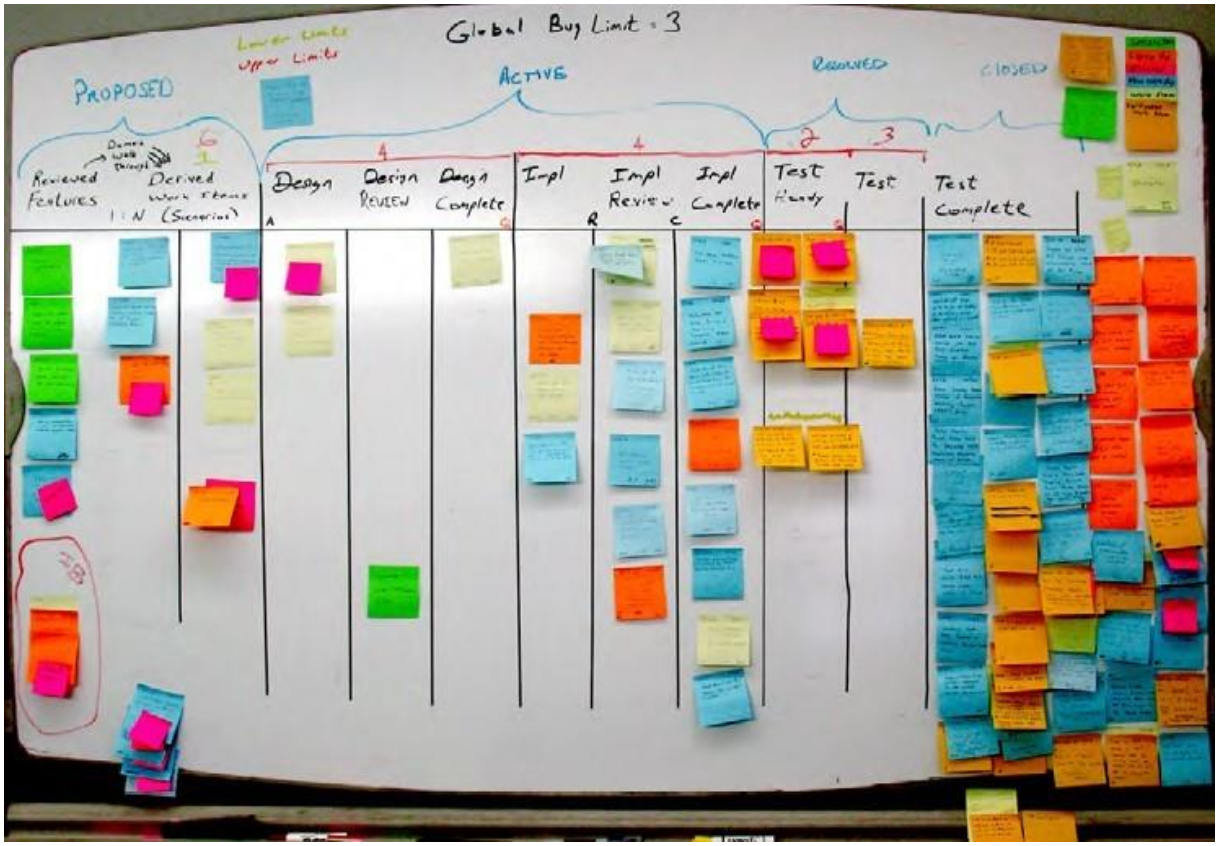

Figura 10 - *Kanban* de uma equipe de desenvolvimento de software. (Fonte: ANDERSON, 2011)

Anderson (ANDERSON, 2011) sugeriu um processo de implementação de *Kanban* na TI, ao qual ele chama de receita para o sucesso. Na sua visão, quando esses elementos são aplicados nas organizações, elas podem obter sucesso no desenvolvimento de software. Os seis passos da receita de Anderson são:

- Foco na qualidade;
- Reduzir Trabalho-em-Progresso;
- Entregar frequentemente;
- Equilibrar a demanda e dendimento;
- Priorizar;
- Atacar fontes de variabilidade para melhorar a previsibilidade.

Com a aplicação do *Kanban* no processo de desenvolvimento de software, as equipes podem se beneficiar da ferramenta para executar os passos propostos por Anderson. A nível de qualidade o *Kanban* permite que cada atividade seja um cartão, dessa forma as tarefas podem ser examinadas e executadas com a devida atenção, além disso, os cartões precisam seguir o

fluxo de trabalho da empresa que é ilustrado no *Kanban*, dessa forma caso o processo tenha etapas de verificação da qualidade (como é esperado que tenha, ver tópico 2.1) garante-se via processo e controla-se pelo quadro que a atividade será verificada e validada para atender os requisitos de qualidade.

O *Work In Progress* (Trabalho em Progresso), mais conhecido como WIP é uma métrica que indica a quantidade de itens em desenvolvimento em um determinado período. Para os times de desenvolvimento que utilizam o *Kanban*, o WIP pode ser facilmente identificado através da quantidade dos itens em andamento, ou seja, itens iniciados e ainda não finalizados. Através do quadro, as equipes podem limitar o valor do WIP, caso o limite da métrica seja atingido nenhuma atividade será puxada enquanto outra não for finalizada. Reduzir o trabalho em progresso, ou encurtar o tamanho de uma iteração, tem um impacto significativo na qualidade (ANDERSON, 2011).

Um dos problemas das metodologias pesadas era o tempo de entrega, por vezes, tornando-se tão longo que resultava em um produto obsoleto, sendo assim, um dos princípios ágeis é “Entregar software funcionando com frequência". Como citado anteriormente, cada cartão finalizado no *Kanban* é uma entrega, este sistema incentiva que cada atividade seja traduzida em um cartão, esse processo tende a quebrar atividades complexas em outras menores, afim de simplificar o trabalho em pequenos passos que podem ser concluídos em tempos mais curtos e por consequência, realizar entregas mais frequentes que tendem a aumentar a confiança das partes envolvidas (ANDERSON, 2011). Além disso, através do quadro há indicação visual de todas as tarefas finalizadas aumentando a motivação do time através da visualização do progresso, além de, agradar as demais partes interessadas que recebem retorno constante do desenvolvimento.

Equilibrar a demanda em relação ao rendimento exige que a organização defina com qual frequência aceitará novos requisitos para desenvolvimento, buscando manter a velocidade no qual as entregas são realizadas (ANDERSON, 2011). O *Kanban* como já citado, por si só é um sistema de produção puxada, quando aliado ao uso do WIP, nenhuma tarefa é puxada enquanto o time não tiver capacidade de executá-la, assim, os times podem ter a segurança de poder manter um ritmo saudável de trabalho. Quando se trabalha dessa maneira, muito do estresse será retirado do ambiente e as pessoas poderão ter a capacidade de focar em cada atividade com precisão e qualidade, elas poderão se orgulhar das entregas e desfrutar do processo (ANDERSON, 2011).

A priorização é feita através da notação adotada pela empresa. Visualmente é muito simples entender através do *Kanban* qual tarefa tem maior prioridade. Cada time pode escolher

ou combinar diferentes elementos para compor sua notação. Esse recurso visual de sinalização, reduz o esforço de sempre ter que definir o que precisa ser feito em seguida, dessa forma as pessoas podem poupar energia desse processo de decisão e aplicá-la para realizar as atividades. Além de tudo, uma priorização clara maximiza o valor da entrega (ANDERSON, 2011).

Pode-se entender como fontes de variabilidade toda a situação interna ou externa que afeta o processo de forma a aumentar, reduzir ou modificar as etapas. Quando existem muitas variabilidades o time não consegue se planejar para manter um ritmo constante, se o time não consegue manter uma constância então todos os passos expostos até aqui serão afetados, uma vez que o time terá que aumentar o esforço para fazer entregas caso a demanda aumente em determinado momento. O *Kanban* auxilia na identificação de variabilidades, é possível identificar situações como:

- Etapas com um acúmulo de tarefas que impedem o fluxo andar de forma fluída (gargalos);
- Etapas onde o trabalho costuma oscilar com amplitude, indo de poucas atividades para muitas atividades em curto período de tempo;
- Etapas onde as atividades constantemente precisam ser adaptadas, incluir uma nova informação ou notação.

Estas situações são indicativos de variabilidade e ao identificar os pontos da cadeia onde elas ocorrem as organizações podem criar planos de melhoria para diminuir o impacto dessas mudanças no fluxo e melhorar produtividade e previsibilidade.

3.2.3 Contribuição

Ao longo dos tópicos deste trabalho, foi evidenciado a dificuldade das metodologias tradicionais de responderem às mudanças em um curto período de tempo. O *Kanban* funciona para este cenário, ele informa de maneira ágil, simples e visual que é necessário uma tomada de ação sobre determinado item. A facilidade com que o *Kanban* executa esse processo de comunicação é diretamente relacionada a como o manifesto ágil enxerga o desenvolvimento do sistema a nível de colaboração e adaptação as mudanças.

Com o uso dos cartões há a indicação do que deve-se fazer a seguir e qual a prioridade disso, de forma a entregar exatamente o que é preciso no momento, pensando em agilidade o *Kanban* é capaz de superar problemas comuns nos modelos clássicos como falta de clareza sobre o trabalho a ser feito e o progresso.

Ademais, o *Kanban* nasceu da necessidade de evitar excessos, pensando em softwares desenvolvidos no modelo tradicional onde havia um excesso de documentação, gerenciamento e regras, o quadro *Kanban* se torna uma ferramenta capaz de agregar característica de uma gestão mais simples e autônoma, uma vez que, os próprios funcionários movimentam o quadro de acordo com os gatilhos específicos, o descritivo do que precisa ser executado é feito no próprio cartão e as regras de utilização são de fácil compreensão. Com essas características o *Kanban* é capaz de substituir outros modelos pesados de gestão.

Este sistema, tem apresentado bons níveis de satisfação do cliente através de entregas constantes, confiáveis e de qualidade. O *Kanban* também colabora nos fatores de prazo e custo, dado que, o sistema controla o fluxo de forma a usar os recursos de acordo com a necessidade e mantendo um ritmo saudável de trabalho, sendo extremamente importante para a qualidade de vida das pessoas envolvidas e trabalho gerado. Ao fazer tudo isso, o *Kanban* contribui para a evolução cultural das organizações, e há sinais que o sistema é um catalisador importante para a formação de uma organização mais ágil (ANDERSON, 2011).

3.3 ***Lean***

O *Lean* emergiu em uma situação de adversidades e escassez enfrentadas no âmbito industrial, ele é composto por um ecossistema de modelos, métodos e práticas, porém ele vai além do processo e pode ser encarado como uma filosofia que prega um pensamento ou mentalidade enxuta, termos que são comumente usados para denominar o *Lean*.

O objetivo fundamental de qualquer empresa é entregar ao cliente o máximo valor possível de forma que obtenha o retorno financeiro para sustentar sua sobrevivência e crescimento. O *Lean* mostra-se competente para esse propósito (HOUSHMAND & JAMSHIDNEZHAD, 2006). Embora tenha surgido em um contexto industrial, o pensamento enxuto não é uma tática da manufatura ou um plano de redução de custos, o *Lean* é uma

estratégia de gestão aplicável a todas as organizações, pois atua nas melhorias dos processos como um todo (WOMACK et al., 2005).

Atualmente, a mentalidade enxuta tem ocupado um espaço importante no mundo corporativo, ela vem apoiando aspectos como competitividade, qualidade, custo, tempo de produção e flexibilidade, o *Lean* tem sido importante para a manutenção da sustentabilidade dentro das organizações (PERALTA et al., 2017).

3.3.1 Histórico

O *Lean Manufacturing* é como o *Lean* é conhecido no ambiente da manufatura e foi o primeiro tipo de modelo *Lean* a surgir. O *Lean Manufacturing* tem suas origens no Japão, no período pós Segunda Guerra Mundial, o país estava assolado pelos conflitos e não dispunha de meios para realizar investimentos expressivos para a implantação da produção em massa proposta pelo fordismo. Para tentar acompanhar o progresso da indústria americana, surgiu a necessidade de se criar um novo modelo gerencial que fosse capaz de dar ao Japão competitividade com poucos recursos (RIANI, 2006).

Taiichi Ohno, executivo da Toyota, ficou responsável por pensar em soluções para tornar a indústria automobilística japonesa competitiva gastando o mínimo e aproveitando o máximo dos recursos. Dos estudos e melhorias propostos por Ohno, nasceu o Sistema Toyota de Produção, os objetivos fundamentais deste novo sistema foram definidos por Ohno da seguinte maneira:

> A eliminação de desperdícios e elementos desnecessários a fim de reduzir custos; a ideia básica é produzir apenas o necessário, no momento necessário e na quantidade requerida. (OHNO, 1997)

Pode-se entender como desperdício, todo trabalho que está presente no processo de produção que aumenta o custo, mas não agregam valor ao produto ou serviço no ponto de vista do cliente (SALGADO et al., 2009). São estas atividades que o STP tem como alvo para serem eliminadas.

Na manufatura foram descritos por Ohno, sete tipos de desperdícios: Superprodução, Espera (Tempo Ocioso), Estoque, Transporte (Transporte desnecessário), Processamento

(Processos obsoletos), Movimento (Movimentação desnecessária de pessoas) e Defeito, quaisquer atividades que possam ser enquadradas em uma dessas categorias, devem ser eliminadas para o STP (OHNO, 1997).

3.3.1.1 Sistema Toyota de Produção

Pensar de maneira enxuta é o que move o Sistema Toyota de Produção, portanto, a mentalidade enxuta está intimamente relacionada ao STP. As principais metodologias do Sistema Toyota de Produção utilizadas para se obter aumento da qualidade e redução de desperdícios de recursos podem ser representadas como uma estrutura delineada (LIKER & MORGAN, 2006).

Para entender a formação do STP, Liker (LIKER & MORGAN, 2006) representa o sistema como uma casa (figura 11), segundo o autor, o porquê da casa é que ela é um sistema estrutural. A casa só é forte se o telhado, as colunas e as fundações forem fortes. Neste modelo de Liker, o telhado representa o objetivo do sistema, as colunas são os pilares que sustentam o sistema e a base é a filosofia que é o que fundamenta toda a estrutura e a mantém firme em seu propósito,

Figura 11 - Estrutura do Sistema Toyota de Produção (Fonte: Adaptado de LIKER e MORGAN, 2006)

O objetivo do sistema é produzir o produto de maior qualidade, com o menor custo possível e com o *Lead Time* mais curto. *Lead Time*, em tradução livre significa Tempo de Espera, entretanto, o conceito de *Lead Time* pode ser entendido como tempo do ciclo de produção, ou seja, o tempo necessário que um produto leva para ficar pronto.

Para alcançar o objetivo, o STP se utiliza dos sistemas *Just In Time* e *Jidoka*. O *Just In Time*, já descrito neste trabalho (ver tópico 3.2.1.1), incentiva a produção na hora certa, o JIT se relaciona ao objetivo de forma a ofertar o produto no menor tempo e utilizando apenas os recursos essenciais sem gerar desperdícios.

Jidoka é uma palavra japonesa para autonomação, que significa automação com um toque humano. A ideia é permitir as máquinas e operadores a capacidade de detectar desvios e parar a produção, o objetivo é evitar que ocorrência de condições anormais levem a produção de produtos defeituosos, neste caso o *Jidoka* está intimamente ligado ao controle de qualidade.

Na base do STP, existem o *Heijuka*, Trabalho Padronizado e *Kaízen*. A finalidade do *Heijuka* é a ação de nivelar o fluxo de produção para corresponder à demanda no longo prazo, nivelar a produção é buscar um fluxo harmônico a fim de refrear ondas de reação frente às oscilações de demanda. A Toyota realiza esse nivelamento através da produção diária de um

conjunto de produtos em pequenas quantidades, assim há sempre uma porção de produtos prontos para atender a demanda, isso evita que uma alta variação na demanda, desestruture a rotina da fábrica, manter a rotina é importante para evitar que o estresse produza excessos e desgastes na produção e nos colaboradores.

Para o STP, o *Kaízen* é um pensamento que prega a busca diária para melhorar os processos com o objetivo de reduzir custos e aumentar a produtividade. No *Kaízen* as pessoas são os agentes de mudança, ele acredita que melhorias ocorrem a partir do pressuposto que as pessoas podem melhorar continuamente no desenvolvimento de suas atividades, afinal as pessoas que estão diariamente trabalhando em suas atividades são os melhores indivíduos para identificar onde há problemas. Neste pensamento, todos os colaboradores devem estar envolvidos na criação de uma mentalidade pautada nos valores de: espírito de equipe, sabedoria, moral e autodisciplina. Esses valores são capazes de criar um terreno propício para novas ideias que possam ser aplicadas em prol do STP (ENDEAVOR, 2020).

Ao lado do JIT e *Kaízen*, temos o Trabalho Padronizado. Ele é um conjunto de procedimentos específicos realizados pelos colaboradores durante o processo de produção. O Trabalho Padronizado é o que mantém a linha de produção funcionando da maneira esperada, uma vez que, o processo é documentado e controlado de acordo com os padrões, eliminam-se as variações que possam vir a causar desperdícios. O Trabalho Padronizado é produto de melhoria contínua (*Kaízen*) e também insumo para que posteriores melhorias possam ser realizadas (*Lean* Institute Brasil, 2020).

Por fim, como visto na figura 11, o STP é sustentado pela a Estabilidade Básica, em resumo, a Estabilidade Básica implica em manter a previsibilidade e disponibilidade constantes em relação a mão-de-obra, máquinas, materiais e métodos, conhecidos como os 4M’s. A motivação é simples, sem a capacidade de garantir os itens fundamentais para produzir aquilo que se deseja, não é possível implementar um fluxo que seja capaz de sustentar o STP (Smalley,2005).

O sistema Toyota se estrutura e trabalha em torno da mentalidade enxuta, a filosofia *Lean Manufacturing* é o resultado da forma de gestão baseada nas práticas e resultados do Sistema Toyota de Produção, sendo o exemplo mais disseminado na busca por processos enxutos (LIKER e MORGAN, 2006).

O *Lean Manufacturing* tem uma filosofia que inspira organizações a adotarem práticas de gestão em busca de otimização do processo produtivo. Portanto, a mentalidade enxuta foi utilizada como alicerce para criação de outros modelos de produção para além da manufatura,

logo a maneira como o *Lean* é personificado depende do setor onde é aplicado. Adiante exploraremos, a aplicação do *Lean* no desenvolvimento de software.

3.3.2 Estrutura

Diante dos resultados expressivos na manufatura, Womarck e Jones lançaram o livro *Lean Thinking* que expunha um pensamento que o *Lean* poderia extrapolar a manufatura e ser aplicado em todos os tipos de setores, de criação de produto ou serviço e que aquilo que demonstrava ser tão bem-sucedido à produção em massa também poderia evoluir para se adaptar a outras realidades (DANTAS, 2016). Womack e Jones foram os responsáveis por popularizar o pensamento ou mentalidade enxuta como uma tradução para captar toda a essência do Sistema Toyota de Produção.

O pensamento enxuto fornece uma maneira de especificar o valor, alinhar as ações de criação de valor na melhor sequência, conduzir essas atividades sem interrupções e executá-las de forma cada vez mais eficaz. Em suma, o pensamento enxuto é enxuto porque fornece uma maneira de fazer mais com menos - menos esforço humano, menos equipamento, menos tempo e menos espaço - enquanto se preocupa em proporcionar aos clientes exatamente o que eles desejam (WOMACK & JONES, 1996).

Existem cinco princípios do *Lean Thinking* que assistem uma gestão organizacional: O valor, a cadeia de valor, o fluxo contínuo, o sistema puxado e a perfeição - melhoria contínua (WOMACK & JONES, 1996):

- Valor: O ponto de partida crítico para o pensamento enxuto é o valor (WOMACK &JONES, 1996). O valor de um produto ou serviço é o que faz o cliente estar disposto a pagar por ele;
- Cadeia de Valor: A cadeia de valor é o processo que o produto ou serviço cumpre até a entrega ao cliente (NETTO, 2017). Durante o mapeamento e acompanhamento da cadeia de valor é possível identificar pontos de desperdício;
- Fluxo Contínuo: O fluxo contínuo configura uma produção sem interrupções com a intenção de atender as demandas do cliente com o menor tempo e custo (NETTO, 2017). A fluidez da produção é importante para o *Lean*, uma vez que a falta dela pode criar gargalos, perda de tempo e trabalho desnecessário.

- Sistema Puxado: A produção puxada, já descrita neste trabalho (ver tópico 3.2.1.1), significa que um processo inicial não deve produzir um produto ou executar um serviço sem que ocorra a solicitação do cliente de um processo posterior (DANTAS, 2016);
- Perfeição (Melhoria Contínua): A perfeição é representada pela melhoria contínua através da rápida detecção e solução de problemas na base (WOMACK et al., 1990). Neste princípio, é esperado que a cadeia de valor seja monitorada e melhorada continuamente até que atinja a perfeição. Apesar de traçar uma meta inatingível, como a perfeição, a verdadeira essência desse princípio é garantir, que mesmo após a implantação inicial do *Lean*, a ideologia tenha continuidade na organização buscando ininterruptamente a melhoria contínua.

A partir do momento que a mentalidade enxuta teve sua aplicação expandida para outras realidades por meio de Womack e Jones, estudos e casos de aplicação surgiram em diversas áreas de conhecimento como: saúde, construção civil, alimentícia e serviços em geral. Em resumo, o *Lean Thinking* possui ênfase na satisfação do que cliente e para alcançar essa satisfação executa práticas aderentes aos cinco princípios (DANTAS, 2016).

Dentre os domínios que o *Lean Thinking* pode ser aplicado, um deles é na Tecnologia da Informação. Dentro da TI, o *Lean Thinking* foi apelidado como *Lean* TI. Nesse modelo, os princípios da mentalidade enxuta são aplicados nos processos de desenvolvimento de software com o intuito de atender a necessidade do cliente, na próxima seção iremos explorar o *Lean* TI.

3.3.2.1 *Lean* TI

Bell e Orzen os pioneiros na exploração do *Lean* TI, define-o como uma transformação comportamental e cultural profunda, que estimula a organização a olhar com uma nova visão para o papel das atividades realizadas na criação e entrega de valor para o cliente. Ainda nessa sequência lógica, então a jornada do *Lean* TI, de acordo com a dupla, deve iniciar nas pessoas e precisa da melhoria contínua, processos e ferramentas/tecnologias nessa ordem (ORZEN & BELL, 2011).

Antes de explorar *Lean* TI, primeiramente, é preciso entender que: desenvolver software é lidar com incertezas. Logo, apesar de semelhanças também há uma diferenciação entre *Lean*

Thinking (base do *Lean* TI) e o *Lean Manufacturing*. O *Lean* TI compartilha a mesma visão do *Kaízen* (Ver tópico 3.3.1.1), em relação à participação protagonista das pessoas no aprimoramento do processo, portanto o primeiro passo da implementação é a mudança de mentalidade para que a cultura *Lean* escorra para o restante da cadeia.

Na manufatura o produto final é conhecido, diferentemente do desenvolvimento de software, portanto aspectos como matéria-prima, material, máquinas, sequência de atividades, mão de obra necessária podem ser previstos com maior facilidade. Á vista disso, faz sentido que os processos e as tecnologias não estejam no início da prioridade do *Lean* TI, pois estes elementos são descobertos durante o progresso das atividades, por isso que as pessoas precisam estar treinadas no pensamento enxuto para que possam ir desenhando o melhor processo no *Lean* TI, e por fim, serem capazes de escolher as ferramentas/tecnologias que apoiem esse processo.

Já um ponto que o *Lean* TI tem em comum com a manufatura é a eliminação dos desperdícios. Entretanto, os desperdícios da TI são diferentes dos encontrados na manufatura. Como desperdícios podemos citar: defeitos, retrabalho, tarefas manuais, perda de conhecimento e informação, atrasos na comunicação, requisitos ultrapassados, hardware subutilizado, dados redundantes, indisponibilidade de recursos, sobrecarga do sistema, entre outros problemas e elementos encontrados no desenvolvimento de software que não agregam valor.

Para identificar e eliminar esses desperdícios, o *Lean* TI se apoia em seus princípios. Os princípios do *Lean Thinking* também receberam uma nova interpretação para o domínio da TI, o *Lean* TI não estabelece um processo, ele apenas direciona a organização para alcançar a mentalidade enxuta. Logo, a sua implementação requer que as organizações tenham, adote ou crie um modelo de trabalho base. Existem alguns modelos que funcionam bem com o *Lean*, mas como destaque podemos citar o *Scrum* e *Kanban*.

Vale ressaltar, que as novas interpretações dos princípios do *Lean* possuem uma relação direta com os valores e princípios ágeis. Neles podemos perceber as preocupações de entrega de valor, satisfação do cliente, entregas constantes, conceito de incremento e melhoria contínua encontradas na agilidade. Abaixo a nova versão dos princípios será detalhada juntamente de sugestões de abordagens de como eles podem ser aplicados no desenvolvimento de software dentro da organização.

- Valor: O *Lean* TI se importa mais com o valor que qualquer outro elemento e todas as atividades devem ser voltadas para produzir produtos ou serviços de valor para satisfazer o cliente. Empresas que desejam implementar o *Lean* TI, podem usar o *Scrum* e *Kanban* uma vez que os dois sistemas têm como foco entregar o que é prioridade;

- Fluxo de Valor: Neste princípio além do controle do processo de desenvolvimento, no *Lean* TI também há a preocupação de identificar o que tem mais valor e o que deve ser entregue primeiro. Pensando em termos de aplicação, no *Scrum* há eventos e artefatos específicos para garantir que as atividades estejam priorizadas e aderente a meta, já o *Kanban* utiliza uma notação para informar a todos a prioridade das atividades;
- Fluxo Contínuo: A definição de Fluxo contínuo é a mesma do *Lean Thinking*, todavia há uma característica que diverge no *Lean* TI, pode ser que para diversas áreas de atividade, como na manufatura, quanto mais fluído e sem interação é o processo menos atrito há entre as partes, logo há uma redução de dependência e eliminação de gargalos.

  Já do ponto de vista do *Lean* TI, as interações são bem-vindas, conforme o valor ágil "Indivíduos e interações mais que processos e ferramentas" para o desenvolvimento de software é importante que as partes interajam, nesse meio a colaboração é essencial para um resultado de qualidade. Desenvolver software é uma atividade intelectual e as interações favorecem esse processo criativo, cria conexões e no entendimento do *Lean* TI gera um fluxo fluído de ideias e progresso.

  Outro ponto de destaque é que para que exista um Fluxo Contínuo no desenvolvimento de software, é preciso uma participação mais ativa do cliente, portanto é recomendado que ele entenda o modelo de trabalho (DANTAS, 2016). O *Scrum* e *Kanban* trabalham bem essas interações quando colocam as pessoas com os agentes de ação e criam espaços para colaboração;
- Produção Puxada: O time deve desenvolver o que o cliente deseja (DANTAS, 2016). Na TI, o princípio de produção puxada é manifestado através da entrega de pequenos pacotes, dessa forma o cliente pode validar continuamente o trabalho, evitando desperdícios e criando um canal para recolher *feedbacks* do mesmo e manter o desenvolvimento apenas no que é essencial. Esse tipo de entrega acontece nos modelos de software que usam como base o processo incremental, como o caso do *Scrum* e *Kanban*, que fatiam suas tarefas em pequenas ações;
- Perfeição: A melhoria contínua faz parte dos princípios da agilidade e ela só acontece no *Lean* TI quando as equipes estão dispostas a inspecionar constantemente o seu trabalho. Para aplicação desse princípio os times devem estar inclinados a manter visível os erros e problemas e as partes interessadas devem contribuir constantemente para o alinhamento do trabalho. No *Scrum*, por exemplo, todos os eventos são oportunidades de adaptar e inspecionar o trabalho de forma a torná-lo cada vez mais perfeito, no *Kanban*

o próprio sistema já é por essência um sistema sinalizador das necessidades mais latentes e como parte do JIT está alinhado a trabalhar com o pensamento de melhoria contínua.

Como visto nos parágrafos anteriores, o *Lean* TI permite que modelos e ferramentas sejam absorvidos pela organização para que alcance seus objetivos. Deste ponto de vista, o *Lean* TI pode ser percebido mais como uma metodologia do que apenas um modelo. Ele está fortemente acoplado com os valores e princípios ágeis e é capaz de absorver e derivar outros modelos de desenvolvimento de software. Entretanto, como ele é uma abstração da metodologia *Lean*, neste trabalho ele é entendido com um modelo dessa metodologia

3.3.3 Contribuição

O *Lean* é valorizado principalmente diante de cenários de crise econômica em que há queda de demanda e necessidade de produção sob demanda (AZEVEDO, 2014). Dois pontos muito comuns do universo da Tecnologia da Informação, pois os softwares possuem um período curto de deterioração, atuam em um mercado consumidor volátil que está sempre à procura da próxima novidade e no geral precisam possuir um nível de personalização alto para atender as demandas do público. Portanto, o pensamento enxuto proporciona a TI ser efetiva com o menor custo, isso é importante porque diante de um domínio tão dinâmico longos investimentos podem não ter tempo hábil para surtir efeitos.

Outra grande contribuição do *Lean* na TI, está na formação de uma nova mentalidade. Como disse Peter Drucker - o pai da administração - " A cultura come a estratégia no café da manhã", o *Lean* aposta em criar uma cultura fortalecida de que seja capaz de movimentar e alimentar o sistema por si mesma. Além disso, por ter base na transformação cultural o *Lean* promove melhorias contínuas e incrementais, dessa forma não são necessárias mudanças dramáticas durante sua implementação.

Ainda em relação aos ganhos da TI com o *Lean*, o objetivo principal do sistema é a eliminação do desperdício, na área de TI, devido às mudanças constantes pode-se criar um software como muitos resíduos de desenvolvimento ou recursos inutilizados, como por exemplo linhas de código obsoletas e Interfaces de Programação de Aplicações (API) inutilizadas, neste

ponto a aplicação do *Lean* pode auxiliar as organizações a manterem processos que consigam absorver mudanças e melhorias sem gerar retrabalho, perdas ou conturbar o processo.

Por fim, uma também valorosa entrega do *Lean* para os negócios é a sua proatividade frente ao mercado. O *Lean* está constantemente buscando a perfeição em toda a cadeia de desenvolvimento, esse movimento dá às empresas que utilizam a vantagem de serem aquelas que lançam as inovações, ao invés de esperar as situações para se adaptarem. Com a aplicação do *Lean* as organizações podem se tornar realmente competitivas e entregar inovações tecnológicas.

O pensamento enxuto fornece uma maneira de tornar o trabalho mais satisfatório (WOMACK & JONES, 1996). O *Lean* não pensa em partes isoladas do processo, ele pensa na organização como um todo e preza para que ela funcione integrada. Womack e Jones em seu livro, expõem exemplos da aplicação da filosofia em diferentes organizações, os exemplos são suficientemente amplos e os resultados tão surpreendentes que não há um executivo que possa alegar que os princípios enxutos não podem ser aplicados a sua empresa

3.4 **Encerramento do Capítulo**

Para o desenvolvimento de software, as incertezas são elevadas, pois muitas vezes se começa um sistema sem ao menos saber ao certo qual será a saída obtida (MCMANUS et al., 2005). Neste capítulo exploramos as particularidades do *Scrum*, *Kanban* e *Lean* no âmbito do desenvolvimento de software e como cada um lida com as incertezas do trabalho. No próximo capítulo, será realizada uma comparação de cada abordagem em relação a critérios significativos para a produção de sistemas.

4 **ANÁLISE COMPARATIVA**

Os modelos ágeis de desenvolvimento de software transformaram a forma como os projetos são executados e são cada vez mais presentes no dia a dia das organizações. Embora existam diversos modelos para atender o desenvolvimento de software, todos possuem vantagens e desvantagens que devem ser consideradas na hora da sua adoção por parte das empresas. No capítulo atual deste trabalho será detalhado um comparativo entre *Scrum*, *Kanban* e *Lean* TI através de critérios relevantes para a agilidade, negócios e times. Primeiramente os critérios serão apresentados, no detalhamento dos mesmos será exposta a motivação da escolha, além das questões que nortearam a análise comparativa. Em seguida, os modelos serão comparados de acordo com os critérios, analisando os ganhos e perdas em relação aos questionamentos levantados. O método comparativo é baseado na experiência da autora em trabalhar com projetos ágeis e gerenciamento de processos em empresas focadas em produtos próprios e consultorias de desenvolvimento de software.

4.1 **Impacto da Implantação da Agilidade**

A cultura organizacional é sustentada e moldada pelas pessoas, neste ponto a agilidade é exigente com a participação e colaboração dos indivíduos, logo no momento que uma organização pondera sobre a adoção de algum modelo ágil, deve primeiramente considerar que mudanças culturais acontecerão, portanto, a empresa precisa estar aberta e ser flexível diante das transformações.

Outro ponto de atenção é que cada modelo ágil, quando aplicado ao desenvolvimento de software, busca atingir objetivos específicos, além disso, existe a possibilidade em muitos casos de combinar duas ou mais abordagens (SOARES, 2019). Portanto, as empresas precisam estar atentas aos objetivos que desejam alcançar através da agilidade e de acordo com as metas escolherem qual ou quais modelos seriam mais adequados.

Por isso, ao longo deste capítulo *Scrum*, *Kanban* e *Lean* TI serão inspecionados a fim de entender como sua implantação pode prover ganhos e perdas para os ambientes aos quais são inseridos. Isto será feito através do destaque de alguns ganhos e perdas mais expressivos de cada

abordagem e como elas se relacionam com critérios significativos para o desenvolvimento de software.

4.2 **Critérios de Comparação**

A seleção dos critérios escolhidos para a comparação foi feita através das perspectivas: Agilidade e Negócio, e serão descritas a seguir juntamente com os questionamentos que serão avaliados na visão de cada abordagem.

4.2.1 Agilidade

O conceito de agilidade ofereceu uma nova visão sobre o desenvolvimento de software. Todavia, para se beneficiar das vantagens é necessário que os processos adotados estejam aderentes aos valores e princípios ágeis. Aspectos importantes para a filosofia ágil, foram considerados como uma forma de verificar o alinhamento entre os modelos e a agilidade. Os critérios selecionados oriundos da agilidade foram:

- Entrega: Como as entregas são feitas dentro de cada model;
- Pessoas: Qual é o papel das pessoas dentro das abordagens;
- Melhoria Contínua: Como os modelos se relacionam com a melhoria contínua;
- Processos e Ferramentas: Como esses elementos se relacionam e compõem os modelos;
- Documentação: Como os modelos tratam a documentação;
- Qualidade: Qual é o papel da qualidade na metodologia;
- Satisfação do Cliente: Como a satisfação do cliente é percebida e garantida.

4.2.2 Negócio

Independente da metodologia que siga, os projetos precisam ser gerenciados. Logo, existem aspectos gerenciais que devem ser levados em consideração, o mais expressivo deles é

a Tripla Restrição. A Tripla Restrição descreve três principais forças que todo o projeto precisa balancear: Escopo, Tempo e Prazo.

A qualidade de “restrição” refere-se à relação conflitante dessas forças. No início do projeto elas são balanceadas através dos contratos iniciais, a tripla restrição é comumente representada como um triângulo em que cada força fica em um vértice, entretanto, conforme o projeto evolui qualquer mudança em um dessas questões irá impactar nas demais. Exemplo, quando se aumenta o escopo do projeto é evidente que se há mais coisas para serem feitas também será necessário mais tempo, mais tempo consequentemente significa mais no custo, já que as pessoas e recursos envolvidos no projeto serão requeridos por mais tempo.

Figura 12 - Tripla Restrição (Fonte: Adaptado do PMOK, 2018)

Ainda no plano da Tripla Restrição (figura 12), existe uma evolução que inclui a qualidade como quarto elemento, a qualidade fica no centro do triângulo representando o objetivo do que as forças precisam alcançar enquanto tentam conciliar interesses. Por isso, entender como os modelos apresentados lidam com essas questões é um fator que deve ser avaliado pelas organizações.

Ainda, na esfera de negócios também será avaliado qual o esforço para implementar e em quanto tempo os resultados são obtidos, afinal a adoção de um novo modelo de trabalho também é um projeto que está sendo executado pela organização. Abaixo os critérios selecionados na esfera de negócios:

- Escopo: Como as abordagens lidam com as mudanças;
- Custo: Dentro deste critério serão avaliadas duas dimensões. A primeira é quais são os possíveis custos de implantação dos modelos nas organizações e a segunda é sobre como os modelos lidam com o custo dos projetos;

- Prazo: Como definem e atendem os prazos dos projetos;
- Implementação e Resultados: Quais são os impactos da implementação na organização e em quanto tempo os resultados são visualizados.

4.3 **Análise Comparativa**

4.3.1 Entrega

O *Scrum*, *Kanban* e *Lean* TI trabalham com entregas incrementais, todos utilizam uma unidade pequena de trabalho, o incremento, que corresponde a uma tarefa a ser executada. Neste modelo de trabalho o cliente sempre recebe o produto esperado ou parte dele. Já no quesito tempo de entrega, o *Scrum* diverge do *Kanban* e *Lean* TI, no primeiro há entrega acontece ao final de um período definido (*Sprint*), já no segundo e terceiro não há definição de um tempo específico para haver uma entrega, contudo todos prezam por uma constância nos ciclos de entrega.

Todavia, é importante que as entregas não sejam apenas o ato de entregar algo novo, mas sim entregar algo de valor, os três modelos buscam entregar aquilo que tem mais valor para o cliente no momento, no entanto isso exige que os times possuam um processo de descoberta do que é prioridade, o *Kanban* faz isso por meio da notação adotada para os cartões já o *Scrum* e *Lean* TI precisam realizar atividades de priorização durante o andamento do desenvolvimento.

4.3.2 Pessoas

Todos os modelos possuem como característica em comum a colaboração. As pessoas passaram a assumir um papel mais participativo e decisório diferente dos modelos tradicionais. A nível de colaboração, o *Scrum* e o *Lean* TI são extremamente colaborativos, eles exigem constantemente que as partes envolvidas participem e deem *feedbacks* sobre o que está sendo produzido e como está sendo produzido.

Já no *Kanban*, o quadro é instrumento de interação e acompanhamento do trabalho em progresso, entretanto, o modelo não explicita rotinas de interação como caso do *Scrum* e *Lean* TI. Mas apesar disso, o *Kanban* oferece bastante autonomia aos colaboradores já que eles são os responsáveis pela movimentação e atualização do quadro.

Os três modelos trazem as pessoas para o centro do processo, isto resulta em empoderamento e motivação para os envolvidos que se sentem parte e responsáveis pelo resultado. Todavia, para que as pessoas se tornem protagonistas dos projetos desenvolvidos é necessário que a organização selecione pessoas qualificadas que sejam capazes de realizar o trabalho esperado e tenham maturidade para seguirem de forma organizada e disciplinada. Caso a empresa não disponha de pessoas com tal perfil é necessário buscar profissionais ou serviços especializados para implatar os modelos.

4.3.3 Melhoria Contínua

Scrum, *Kanban* e *Lean* TI possuem valores ancorados na melhoria contínua, eles se preocupam constantemente em inspecionar o processo em busca de pontos de otimização. No entanto, os modelos não definem métodos para descoberta de melhoria contínua, cabendo a organização buscar ferramentas que melhor se encaixam em sua realidade para esse tipo de investigação. No *Kanban* não há definição de uma rotina de melhoria contínua como no *Scrum*, contudo o *Kanban* é um sistema no qual é possível identificar pontos de desvios, o quadro auxilia os times a encontrarem seus gargalos no processo.

O ato de melhorar continuamente o processo consome tempo de desenvolvimento, entretanto permite que as empresas evoluam seus produtos e serviços. A melhoria contínua é um elemento importante para que a aplicação de modelos ágeis seja bem-sucedida, a falta dela pode levar ao fracasso da implementação da agilidade, por isso, o tempo gasto em melhorias deve ser contabilizado também como parte da produção, afinal, elas são atividades importantes para agregar valor ao produto ou serviço final.

4.3.4 Processos e Ferramentas

Os modelos escolhidos possuem relações distintas em relação a processo e ferramentas. O *Kanban* é um sistema onde o quadro é a ferramenta que vai auxiliar os times no desenvolvimento de software, contudo, para que o quadro funcione é básico que a organização já tenha um processo estabelecido.

O *Scrum* define um processo de trabalho que precisará ser adotado pela empresa, todavia, ele deixa aberto para que as instituições utilizem métodos auxiliares para apoiar o processo. Um ponto de atenção é que para algumas organizações pode ser difícil escolher por si só os métodos de apoio, o que pode levar a escolhas equivocadas que acabam atrapalhando o *Scrum* e adicionando burocracias.

O *Lean* TI não define um processo, contudo os valores da abordagem devem ser utilizados para criar um processo com a essência *Lean Thinking*. Apesar de estabelecer princípios, cabe à empresa traduzi-los a sua realidade, o que pode dar bastante liberdade para os negócios se adaptarem, mas como *Scrum*, também pode tornar o conceito abstrato para algumas organizações, dificultando o entendimento. Quanto a ferramentas, o *Lean* TI oferece uma diversidade de práticas para implantar o pensamento enxuto, o próprio quadro *Kanban* é uma delas, porém existem outras práticas como o *Kaízen* e o *Jidoka*.

4.3.5 Documentação

Nenhuma das formas de trabalho detalhada como lida com a documentação dos seus processos, mostrando que a criação de documentação não é prioridade. No *Lean* TI, o que se espera é a incessante busca pela melhoria contínua e isso só pode acontecer quando o processo está visível, mapeado e controlado, portanto apesar de não expor diretamente o uso de documentação, ela precisa de uma certa forma existir para que ocorra o acompanhamento do processo.

No *Scrum* e *Kanban* a documentação também tem um papel secundário no processo, assim como o *Lean* TI, nenhuma das duas formas deixa claro como a documentação é executada, no entanto, os artefatos utilizados nesses modelos de certa forma atuam como o papel de

documentação. No *Scrum*, o *Backlog* do Produto e o *Sprint Backlog* são planos de trabalho utilizados pela equipe para saber o que fazer a seguir e constantemente estão sendo revisados. No *Kanban*, o cartão contém as informações do que precisa ser feito e também pode conter explicações que auxilie o time na execução.

Também é possível ter a documentação do projeto nas próprias ferramentas que apoiam a execução do modelo de trabalho, como por exemplo sistemas de gerenciamento de projetos baseados nos modelos ágeis, sendo assim a própria descrição da tarefa se torna a documentação do projeto.

4.3.6 Qualidade

O valor é o espírito da agilidade, todas as práticas, processos, métodos são voltados para entregar valor. Intimamente ligada a esse objetivo existe a qualidade do produto ou serviço, pois além de entregar o que se é esperado no momento certo, é também essencial que o produto ou serviço atenda o cliente sem falhas. A qualidade é expressada em diversos pontos do processo nos modelos ágeis.

No *Scrum* temos as inspeções regulares, eventos com *feedback*, Definição de Pronto, entre outras práticas que visam agregar valor. No *Lean* TI o princípio da perfeição traduz o nível de qualidade esperada dos resultados de uma organização enxuta, portanto são utilizados assim como no *Scrum* pontos de inspeção da qualidade, porém no *Lean* TI a responsabilidade fica concentrada nas pessoas que devem ser treinadas para identificar desperdícios e melhorias.

Já no *Kanban* espera-se que a organização tenha uma fase do seu processo que seja responsável por garantir a qualidade do que está sendo produzido. O *Kanban* também se utiliza de estratégias como WIP para limitar o trabalho com a intenção de promover um ritmo de trabalho sustentável e que mantenha a qualidade dos produtos do trabalho.

A preocupação com a qualidade pode apresentar um impacto no prazo do projeto, uma vez que agilidade, não tem relação com entregar mais rápido, mas sim ser ágil para se adaptar às mudanças. Logo, organizações que acreditam que adotar um modelo de trabalho ágil irá necessariamente levar a uma diminuição de prazo, provavelmente estão enganadas, exatamente porque as diversas inspeções consomem tempo de projeto e quanto mais crítico o software mais inspeções necessitará.

4.3.7 Satisfação do Cliente

Todos os passos, princípios e valores adotados pelos modelos ágeis tem um único objetivo: Satisfazer o cliente. Neste ponto, os três modelos compartilham da mesma estratégia, a entrega contínua de valor. Eles buscam através do fluxo contínuo garantir que o cliente tenha insumos para fornecer *feedback* sobre o que está sendo oferecido, dessa forma os modelos percebem o quão valoroso é a entrega e os pontos de melhoria.

Uma questão que exige cuidado é que a entrega constante exige um gerenciamento das expectativas do cliente. Os clientes podem trazer diversos pontos de melhoria sobre um produto ou serviço, dependendo da quantidade de clientes e a diversidade dos mesmos, essas solicitações tendem atingir um grande volume e variedade, isso pode se tornar um problema para as empresas que possuem um processo fraco de priorização.

No *Scrum* existe a figura do *Product Owner* que é responsável pode gerenciar essas solicitações e priorizá-las, já no *Lean* TI e *Kanban* esse papel não existe, logo, cabe às instituições criarem esse papel ou adotar um conjunto de passos que auxiliem os times a priorizar o que é importante. No *Kanban* essa notação já é esperada, mas como dito anteriormente ela deve partir de um consenso dos indivíduos.

Contudo, por mais que diversas solicitações possam ser difíceis de gerenciar elas são muito bem-vindas nos três modelos, são através desses *feedbacks* que as organizações conseguem compreender seu cliente e perceber quais são suas reais dores, dores essas que serão sanadas através do produto ou serviço oferecido.

4.3.8 Escopo

O escopo é um dos pontos críticos do desenvolvimento ao lado de custo e prazo. Nos modelos analisados todos são abertos a alterações de escopo, pois eles entendem que as mudanças fazem parte do desenvolvimento de software. No *Lean* TI e *Kanban* as mudanças podem ocorrer a qualquer momento do processo e isso traz muita flexibilidade a produção, porém é preciso ter prudência, pois mudanças extremamente constantes produzem o efeito

contrário levando os times a nunca conseguirem alcançar um resultado pois as prioridades estarão sempre mudando.

A fim de contornar o problema citado, o *Scrum* não encoraja mudanças ao longo da *Sprint*, uma vez que o que precisa ser produzido é definido no início do ciclo e não deve ser alterado, portanto, cabe ao cliente esperar o início de uma nova *Sprint* para incluir mudanças. Apesar de exigir que certas tarefas sejam finalizadas antes de partir para uma mudança, essa forma de trabalho evita que o trabalho seja descartado antes de ser inspecionado pelo cliente.

4.3.9 Custo de Implementação

Ao que diz respeito aos custos de implementação, os modelos podem divergir consideravelmente. O *Kanban* pode ser considerado o mais barato de ser implementado, dado que, os recursos utilizados são o quadro e cartões, fora isso não há necessidade de adesão de outras ferramentas e o processo pode seguir o fluxo já estabelecido, não exigindo grandes mudanças na fase inicial.

Já o *Scrum* exige um custo de implementação maior dependendo do que a organização já dispõe, pois apesar de não determinar ferramentas, o *Scrum* precisa de pessoas qualificadas para realizar o trabalho e que exerçam os papéis estabelecidos pelo modelo de trabalho. Se a instituição não possuir esse perfil de funcionário, será necessário realizar contratações ou aplicar treinamentos para capacitar os colaboradores existentes. Como o modelo *Scrum* não varia nas suas regras, uma vez aprendido pode ser mantido com uma maior facilidade.

Dentre os três o *Lean* TI é o com o maior custo de implementação, além de pessoas qualificadas, para que o *Lean* realmente seja utilizado da forma correta é necessária uma mudança de mentalidade, portanto todos os colaboradores precisam receber constantes estímulos para pensar na redução de desperdícios, além disso, também há o custo de manutenção de recursos para que os mesmos não sejam fonte de desperdícios e funcionem de forma eficiente.

Apesar do *Lean* TI buscar uma produção sem desperdícios e com o menor custo possível, o modelo exige uma manutenção maior em relação aos outros modelos, não necessariamente que esse custo de manutenção seja todo em forma de dinheiro, mas há um custo de tempo e capital intelectual para mover o sistema em direção a perfeição.

4.3.10 Custo do Projeto

O *Scrum* e o *Kanban*, diferente do *Lean* TI, não adotam uma política tão clara sobre custo de produção, é certo que há uma preocupação com o custo, mas ela é percebida através da entrega contínua, uma das exigências que essa continuidade busca atender é não desperdiçar recursos produzindo algo não esteja aderente ao desejo do cliente, entretanto se esse desejo representar um aumento do custo, mas o cliente está de acordo com adição, o *Scrum* e *Kanban* atendem essas requisições com certa facilidade. Mais uma vez, é preciso ter atenção a priorização, ficar à mercê da imaginação das suas partes interessadas que por muitas vezes puxam o desenvolvimento para visões divergentes gerando impactos no custo.

Quanto ao *Lean* TI, esse tem uma visão clara sobre custo e desperdício, o modelo preza pela redução de qualquer elemento que não agregue valor, logo a diminuição do custo é uma variável importante e com devido destaque. Diferente dos dois modelos anteriores, o *Lean* TI sempre almeja a produção ao menor custo, logo mesmo que as alterações precisem ser realizadas, empresas *Lean* TI devem fazê-las da maneira inteligente e não de forma reativa, em razão de que, alterações prematuras no processo tendem a ser onerosas.

4.3.11 Prazo

O *Lean* TI e *Kanban* não determinam prazos, ambos prezam pela entrega contínua, porém o ritmo de entrega vai depender de fatores externos ao modelo, como por exemplo a demanda e acordos firmados em contratos. Nessas duas abordagens a organização precisa aprender sobre si mesma para determinar um fluxo de entrega que seja capaz de atender.

Para adoção do *Lean* TI e *Kanban* a empresa precisa ser madura para conseguir criar mecanismos de medição da sua velocidade de execução, o próprio quadro *Kanban* ajuda a identificar esse fluxo de acordo com a quantidade de itens que são executados. O *Lean* TI pode utilizar o conceito de *Takt Time*, oriundo da manufatura. *Takt Time* é o cálculo do tempo de trabalho subtraindo o tempo de atividades que não estão ligadas diretamente a produção como pausas, refeições, treinamentos, entre outras rotinas.

Na questão de prazo o *Scrum*, utiliza a *Sprint* para garantir que ao final de períodos definidos haverá uma entrega, essa estratégia oferece as partes interessadas a segurança de um prazo fixo, mas também permite os times a se organizarem da maneira que preferirem durante o ciclo, tirando a pressão do questionamento comum "Quando ficará pronto? ". Apesar de existir o evento *Sprint*, a quantidade de Sprint*s* necessárias varia de acordo com cada projeto, logo assim como o *Lean* TI e o *Kanban* o prazo total de projetos *Scrum* também depende de questões externas ao processo.

4.3.12 Implementação e Resultados

O Guia *Scrum* define o modelo como leve, fácil de aprender e difícil de dominar. O *Scrum* tem sido utilizado tanto no contexto da Tecnologia da Informação, quanto em setores industriais, educacionais, governamentais e diversas outras áreas de produção ou serviços. Por isso, quando o Guia *Scrum* utiliza palavras "desenvolver" e "desenvolvimento" ele está se referindo a trabalhos complexos de qualquer ambiente como os citados anteriormente (SCHWABER & SUTHERLAND, 2017). Logo, fica evidente que este modelo é adaptável a diversas esferas, aumentando sua popularidade e adesão por diferentes instituições.

No quesito aprendizado o *Scrum* é fácil de assimilar as regras e os elementos que compõem o modelo são simples de compreender, apesar de ser atualizado por seus criadores, o Guia *Scrum* não passa por grandes alterações desde sua criação, logo a sua essência permanece inalterada facilitando sua manutenção dentro das empresas.

O *Scrum* é o mais detalhado dos modelos selecionado, comparado aos demais, para que sua implementação produza o resultado esperado é preciso que todos sigam suas recomendações à risca, papéis, artefatos e eventos não podem ser alterados e as empresas precisam ter disciplina na sua implementação. Isso pode gerar um certo estresse na rotina de trabalho vigente, pois o modelo deverá ser implementado integralmente ocasionando uma mudança as vezes radical.

Contudo, esse detalhamento torna mais fácil para as empresas que querem começar a ser ágeis saber quais são e a ordem dos passos a serem seguidos e dessa maneira, os resultados podem ser vistos mais rapidamente, já que o *Scrum* fornece uma estrutura completa de trabalho, o objetivo é que já na primeira *Sprint* o processo gere uma entrega.

Seguindo a análise, apesar da vontade das organizações adotarem a mentalidade enxuta para sua cultura interna, ainda é desafiador para os negócios aplicar essa transformação e medir a sua efetividade (DANTAS, 2016). O *Lean* e suas variações são difíceis de se implementar, essa característica se deve ao fato que para que o pensamento enxuto prospere é necessário que toda a empresa adote essa mentalidade. O *Lean* TI diferente do *Scrum* tem a vantagem de poder ser implementado aos poucos, primeiramente só em algumas áreas diminuindo o estresse organizacional, mas ele só vai produzir o efeito máximo quando toda a organização for uma organização com pensamento enxuto.

O *Lean* como já citado neste trabalho extravasou da manufatura para outros setores, mostrando sua capacidade de ser aplicada a diferentes contextos, contudo, para cada nova área que deseja adotar o *Lean* é preciso que ele seja adaptado, como o caso do *Lean* TI, dessa forma apesar de proporcionar flexibilidade, ele não é tão genérico quanto o *Scrum*.

O aprendizado do *Lean* TI é custoso, como o modelo não define um passo a passo, as empresas precisam se esforçar para aplicá-los na prática e por muitas vezes contratar um serviço externo para auxiliar esse processo, em contrapartida também ganham liberdade para criar suas próprias interpretações com um método menos impositivo. Esse processo de autoconhecimento por parte das instituições é um fator que torna o *Lean* TI um modelo de trabalho com um tempo maior para conceber resultados expressivos e por ter uma aplicação muito subjetiva também dificulta a medição da efetividade, todavia é um modelo que pode proporcionar uma vantagem competitiva a nível de inovação quando amplamente aplicado, o *Lean* é capaz de tirar uma organização de uma zona comum e transformá-la em referência no mercado.

Por fim, o *Kanban* dentre os três modelos é o que apresenta maior facilidade de implementação, a única exigência é que o modelo necessita de um processo já definido como base. O aprendizado do *Kanban* é muito simples e a notação pode ser escolhida colaborativamente pela organização.

O processo com *Kanban* só se movimenta se os colaboradores movimentarem o quadro, essa característica exige que as pessoas tenham bastante disciplina, pois qualquer esquecimento pode criar gargalos ou atrasar o desenvolvimento. O modelo também se adapta a diferentes ambientes e os times podem possuir mais de um quadro *Kanban* paralelamente.

Na implementação do *Kanban* é preciso separar o processo de desenvolvimento da capacidade produtiva. O quadro *Kanban* não interfere na capacidade produtiva da organização, ele apenas sinaliza o progresso do desenvolvimento, diferente dos outros modelos o *Kanban* não traz práticas que sejam capazes de alterar o processo, todavia, ele torna o processo visível.

A transparência que o *Kanban* traz para o processo favorece as empresas a nível de resultado, em poucos dias o *Kanban* é capaz de mostrar com nitidez onde são os pontos fortes e fracos do desenvolvimento, isso permite colocar esforço para solucionar os problemas a fim de alcançar o fluxo contínuo, essa visibilidade proporciona às organizações serem mais assertivas na implementação de melhorias.

4.4. **Tabela de Comparação entre os Modelos Ágeis**

Os critérios e análises descritos na seção anteriores foram organizados na tabela 1 de forma a sintetizar os resultados.

Tabela 1 – Síntese da comparação dos modelos de trabalho (continua)

Modelo Ágil / Critério de Análise	***Scrum***	***Kanban***	***Lean* (*Lean* TI)**
Entrega	Entregas constantes e incrementais ao final do ciclo de trabalho.	Entregas constantes e incrementais.	Entregas constantes e incrementais.
Pessoas	Colaborativo.	Autonomia.	Altamente responsáveis pelo processo.
Melhoria Contínua	Eventos de inspeção e adaptação.	Permite visibilidade, mas não define um método.	Busca da perfeição.
Processos e Ferramentas	Especifica atividades. Ferramentas livres.	Utiliza atividades existentes. Usa quadro como ferramenta.	Não especifica atividades. Ferramentas livres.
Documentação	Usa artefatos como documentação.	Cartão é uma forma de documentação.	Incentiva a identificação, detalhamento e acompanhamento do processo.

Tabela 1- Síntese da comparação dos modelos de trabalho (conclusão)

Modelo Ágil / Critério de Análise	***Scrum***	***Kanban***	***Lean* (*Lean* TI)**
Qualidade	Inspeções regulares e definição de pronto.	Garante via processo o valor dos entregáveis.	Autonomação.
Satisfação do Cliente	*Product Owner* papel responsável por representar as partes interessadas.	Acordos organizacionais para priorizar o que tem mais valor.	Eliminação de desperdício.
Escopo	O escopo só pode ser alterado em uma periodicidade determinada.	O escopo pode mudar a qualquer momento.	O escopo pode mudar a qualquer momento.
Custo da Implementação	Custo baixo de infraestrutura e custo moderado de capacitação.	Custo baixo de infraestrutura e capacitação.	Custo alto de capacitação. Aquisição de infraestrutura é variável.
Custo do Projeto	Aceita variações no custo desde que esteja relacionada a entrega de valor.	Aceita variações no custo desde que esteja relacionada a entrega de valor.	Preocupa-se com o custo focando na eliminação e desperdícios.
Prazo	Fluxo contínuo de entregas, delimitado por contratos.	Fluxo contínuo de entregas, delimitado por contratos.	Fluxo contínuo de entregas, delimitado por contratos. *Takt Time*.
Implementação e Resultados	Modelo de trabalho de difícil domínio. Resultados na primeira Sprint.	Fácil implementação. Resultados dependentes do processo organizacional adotado.	Implementação em etapas. Resultados a longo prazo, mas oferece grande vantagem competitiva.

4.5 **Encerramento do Capítulo**

Neste capítulo, foi possível comparar os modelos e explorar como cada um procura criar um processo capaz de gerar resultados para as empresas. Como pode ser visto, os modelos *Scrum*, *Kanban* e *Lean* TI, apresentam forças e fraquezas durante sua implementação e uso que devem ser consideradas em futuras implementações.

5 **CONCLUSÃO E TRABALHOS FUTUROS**

Não há dúvidas que as metodologias ágeis são atualmente a forma de desenvolvimento de software mais bem-sucedida e popular a nível mundial. É relevante entender as motivações e propostas levantadas por essa nova abordagem de trabalhar.

Este trabalho possibilitou compreender a natureza da agilidade e seus ganhos em relação as metodologias tradicionais. Além disso, foi possível examinar detalhadamente os modelos de trabalho mais populares quando se trata de desenvolvimento de software: *Scrum*, *Kanban* e *Lean* TI.

Através das análises realizadas dos modelos selecionados foi possível traçar uma comparação entre eles e os possíveis resultados que uma organização teria na adoção de uma das três abordagens.

A pesquisa possibilitou gerar questionamento que devem ser considerados pelas empresas e graças a comparação pode ser possível trazer clareza a aspectos poucos considerados na hora de escolher um modelo de trabalho.

Também foi possível observar que nenhum modelo representa uma solução pronta e ideal para as organizações, por isso, é essencial que as empresas não encarem a agilidade com a salvação para todos os problemas de desenvolvimento de software, mas sim como uma forma de adotar um método de trabalho que precisa ser mantido, acompanhado e evoluído constantemente e prol de resultados positivos.

Como sugestão para trabalho futuros, dar continuidade a este trabalho avaliando se os modelos ágeis selecionados ainda são aderentes ao desenvolvimento de software ou se é possível que novas formas de trabalho que estejam emergindo nas organizações e venham a se consolidar como novos modelos ágeis de desenvolvimento de software.

REFERÊNCIAS

ALMEIDA, D. R. (2015). *Sistema Kanban: Revisão Bibliográfica da Ferramenta de Controle de Produção e De Estoques.* 22 f. Trabalho de Conclusão de Curso (Bacharelado em Administração) - Faculdade Damas da Instrução Cristã, Recife. Disponível em: https: //faculdadedamas.edu.br/revistafd/index.php/academico/article/download/862/701. Acesso em: 04 Dez 2020.

ANDERSON, D. (2011). *Kanban - Mudança Evolucionária de Sucesso para seu Negócio de Tecnologia*. Blue Hole Press. Sequim, WA, USA.

AZEVEDO, R. G. (2014). *Aplicação de Princípios do Pensamento Enxuto no Processo de Envio e Devolução de Correspondências em um Banco Privado*. Trabalho de Graduação (Engenharia de Produção Mecânica) - Faculdade de Engenharia do Campus de Guaratinguetá, Guaratinguetá. Disponível em: https://repositorio.unesp.br/bitstream/handle/ 11449/123015/000806231.pdf?sequence=1. Acesso em: 14 Dez 2020.

BARBOSA, M. (2020). *O Que é Scrum e como Executar Suas Tarefas em Metade do Tempo?* Disponível em: https://outboundmarketing.com.br/o-que-e-scrum/. Acesso em: 05 Nov 2020.

BARROS, A. M. G. & JUNIOR, M. L. (2011). *Implantação do Sistema Kanban Eletrônico Externo em Uma Filial do Segmento de Tintas Automotivas: Uma Pesquisa-Ação*. http: //www.abepro.org.br/biblioteca/enegep2011_tn_sto_135_856_17746.pdf.. Acesso em: 04 Dez 2020.

BECK, K., BEEDLE, M., BENNEKUM, A. V., COCKBURN, A., CUNNINGHAM, W., FOWLER, M., GRENNING, J., HIGHSMITH, J., HUNT, A., JEFFRIES, R., KERN, J., MARICK, B., MARTIN, R. C., MELLOR, S., SCHWABER, K., SUTHERLAND, J. & THOMAS, D. (2001). *Manifesto for Agile* Software *Development*. Disponível em: https://agilemanifesto.org/. Acesso em : 12 Out 2020.

BESSA, T. & ARTHAUD, D. D. B. (2018). *Metodologias ágeis para o desenvolvimento de* softwares. Revista Ciência e Sustentabilidade - CeS. v. 4, n. 2, p. 173-213, jul./dez. Disponível em: https://periodicos.ufca.edu.br/ojs/index.php/cienciasustentabilidade/ article/download/314/308/. Acesso em 04 Dez 2020.

CADORIN, F. (2015). *Uma Proposta de Implantação do Sistema Kanban para a Empresa Faimec Móveis*. 60 f. Monografia (Bacharelado em Administração) - Universidade do Extremo Sul Catarinense, Criciúma. Disponível em: http://repositorio.unesc.net/handle/ 1/3914. Acesso em: 05 Dez 2020.

CHAPLIN, C. (1936).Tempos Modernos. United Artists.

DANTAS, C. L. (2016). *Lean IT: Estudo de Lean Thinking na Área de Tecnologia da Informação*. 57 f. Trabalho de Graduação Interdisciplinar (Tecnologia em Análise e Desenvolvimento de Sistemas) - Faculdade de Tecnologia, Limeira. Disponível em: https://liag. ft.unicamp.br/leanit/wp-content/uploads/sites/8/2017/05/5414VfinalTCCCibele.pdf. Acesso em: 09 Dez 2020.

DeGRACE, P. & STAHL, L. H. (1990). *Wicked Problems, Righteous Solutions: A Catalogue of Modern* Software *Engineering Paradigms*. 1ª ed. Nova Jersey:Yourdon Press.

DOMINGUES, H. (2004). *Governança de TI – Um caminho sem volta*. São Paulo: International Business Communications.
ENDEAVOR. (2020). *Kaizen: A Sabedoria Milenar a Serviço da sua Melhor Gestão*. Disponível em: https://endeavor.org.br/operacoes/kaizen/. Acesso em: 19 Dez 2020.

GIRARDI, H. M. (2016). *Kanban em Serviços: Estudo de Caso em uma Empresa De TI* [81 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia de Produção) - Universidade Federal do Espírito Santo, Vitória. Disponível em:]. https://producao.ufes.br/sites/producao.ufes.br/files/field/anexo/2016_kanban_em_servicos_estudo_de_caso_em_uma_empresa_de_ti.pdf. Acesso em: 05 Dez 2020.

HERBSLEB, J. & MOITRA, D. (2001). *Global* Software *Development*. IEEE Software, California, v. 16, n. 2, p. 16-20.

HOUSHMAND, M. & JAMSHIDNEZHAD, B. (2006). *An Extended Model of Design Process of Lean Production Systems by Means of Process Variables. Robotic and Computer-Integrated Manufacturing*. 16 f. Artigo (Engenharia Industrial) - Sharif University of Technology, Irã.

JUNIOR, R. P. P. (2003). *Kanban: Sua Utilização na Indústria, Visando Redução de Custos Através da Organização e Controle de Estoques*. 39 f. Trabalho de Conclusão de Curso (Ciências Contábeis) - a Universidade Federal de Santa Catarina, Florianópolis. Disponível em: http://tcc.bu.ufsc.br/Contabeis300706.PDF. Acesso em: 10 Dez 2020.

KNOWLEDGE 21. (2019). *O que é Scrum?* Disponível em: https://knowledge21.com.br/blog/o-que-e-scrum/. Acesso em: 23 Dez 2020.

KNOWLEDGE 21. (2020). *Série de Postagens sobre os Papéis do Scrum*. Disponível em: https: //knowledge21.com.br/blog/category/papeis/. Acesso em: 11 Nov 2020.

KOSCHEVIC, M. T. (2001). *Gerenciamento de Processos com Metodologias àgeis*. 36 f. Monografia (Pós Graduação Engenharia de Software) - Universidade Tecnológica Federal do Paraná. Disponível em: http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/1120/3/MD_ENGESS_I_2012_15.pdf.. Acesso: 03 Dez 2020.

LARMAN, C. (2003). *Agile and Interative Development: A Manager's Guide*. Addison-Wesley Professional.

LEAN INSTITUTE BRASIL. (2020). *O que é Trabalho Padronizado?* Disponível em: https://www.lean. org.br/conceitos/126/o-que-e-trabalho-padronizado.aspx. Acesso em: 19 Dez 2020.

LIKER, J. & MORGAN, J. (2006). *The Toyota Way in Services: The Case of Lean Product Development*. The Academy of Management Perspectives, v. 20, n. 2, p. 5-20.

LIMA, G. R. D. C. (2015). *Benefícios das metodologias ágeis no gerenciamento de projetos de Tecnologia da Informação (TI)*. 21 f. Artigo (MBA Governança nas Tecnologias da Informação) - Instituto de Pós-Graduação e Graduação, Goiânia. https://docplayer.com. br/16101207-Beneficios-das-metodologias-ageis-no-gerenciamento-de-projetos-detecnologia-da-informacao-ti.html. Acessado em: 02 Dez 2020.

LUBBEN, R. (1989). *Just-In-Time: Uma Estratégia Avançada De Produção*. 2.ed. São Paulo: McGraw-Hill.

LUDVIG, D. & REINERT, J. D. (2007). *Estudo do Uso de Metodologias Ágeis no Desenvolvimento de uma Aplicação de Governo Eletrônico*. 157fl. Trabalho de Conclusão de Curso (Sistema de Informação) - Universidade Federal de Santa Catarina (UFSC), Florianópolis. Disponível em: https://repositorio.ufsc.br/handle/123456789/184380. Acesso em: 04 Dez 2020.

MAHNIC, V. & DRNOVSCEK, S. (2005). *Agile* Software *Project Management with Scrum*. 6 f. Artigo - University of Ljubljana. Disponível em: https://www.researchgate.net/ publication/228967959_Agile_Software_Project_Management_with_*Scrum*. Acesso em: 05/12/2020.

MARÇAL, A. S. C., FREITAS, B. C. C. D., SOARES, F. S. F., MACIEL, T. M. M. & BELCHIOR, A. D. (2007). Estendendo o SCRUM segundo as Áreas de Processo de Gerenciamento de Projetosdo CMMI. Disponível em: https:/ /docplayer.com.br/1330324 - Estendendo- o- scrum- segundo- as- areas- de- processo- de- gerenciamento- de- projetos-do-cmmi.html. Acesso em: 27 Dez 2020.

MARQUES, A. N. (2012). *Metodologias ágeis de desenvolvimento: Processos e Comparações*. 66 f. Monografia (Processamento de Dados) - Faculdade de Tecnologia de São Paulo, São Paulo. Disponível em: http://www.fatecsp.br/dti/tcc/tcc00064.pdf. Acesso: 27 Dez 2020.

MCMANUS, H., HAGGERTY, A. & MURMAN, E. (2005). *Lean Engineering: Doing the Right Thing Right*. 1st. In: International Conference on Innovation; Integration in Aerospace Sciences. Belfast.

MORALES, C. V. G. & COIMBRA, M. M. (2017). *Implementação de Sistema Kanban no Estoque de Matéria-Prima de uma Linha de Extrusão de Limpador de Para-Brisa*. 53 f. Trabalho de Conclusão de Curso (Engenharia Mecânica) - Universidade Tecnológica Federal do Paraná, Curitiba. Disponível em: http://repositorio.roca.utfpr.edu.br/jspui/ bitstream/1/9331/1/CT_DAMEC_2017_1s_18.pdf. Acesso em: 10 Dez 2020.

NETTO, E. L. P. (2017). *Proposta de Melhoria do Processo de Cadastramento de Pacientes Baseada em Conceitos Lean: O Caso de um Centro de Fisioterapia da Região Metropolitana*. Trabalho de Conclusão de Curso (Engenharia de Produção) - Universidade Federal Fluminense, Rio das Ostras. Disponível em: https://app.uff.br/riuff/bitstream/1/ 5911/1/TCC%20EVALDO%20LUIS.pdf. Acesso em: 09 Dez 2020.

OHNO, T. (1997). *O Sistema Toyota de Produção: Além da Produção em Larga Escala*. Porto Alegre: Bookman.

OLIVEIRA, B. C. (2018). *Gaia Protótipo: Um Modelo De Prototipação Para Processos de Desenvolvimento De* Software. 58 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Universidade Estadual de Londrina, Londrina. Disponível em: http://www.uel.br/cce/dc/wp-content/uploads/TCC-BRUNO_CARAZATO_DE_ OLIVEIRA-BCC-UEL-2017.pdf. Acesso: 03 Dez 2020.

OLIVEIRA, J. F. (2014). *A utilização da metodologia Scrum sob a percepção da equipe de desenvolvimento em uma empresa privada de* software*: Um estudo de caso*. 78 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências Aplicadas e Educação) - Universidade Federal da Paraíba, Rio Tinto. Disponível em: https://si.dcx.ufpb.br/wpcontent/uploads/2015/12/Jessyca-Ferreira-de-Oliveira.pdf. Acesso em: 05 Dez 2020.

ORZEN, M. & BELL, S. (2011). *Lean IT: Enabling and Sustaining Your Lean Transformation*. New York: Productivity Press. 370 p.

PERALTA, C. B. D. L., LERMEN, F. H., MATIAS, G. D. S., SILVA, V. L. D., RIBEIRO, G. F. & ECHEVESTE, M. E. S. (2017). *Princípios Lean no Processo de Desenvolvimento de Produto (PDP) – Uma Abordagem Qualitativa*. Revista Espacios, v. 38, n. 27, p. 35-51. Disponível em: https://www.revistaespacios.com/a17v38n27/a17v38n27p36.pdf. Acesso em: 15 Dez 2020.

PEREIRA, P. (2018). *KANBAN – Estudo de Caso em Indústria de Confecção*. 46 f. Trabalho de Conclusão de Curso (Engenharia Mecânica) - Universitário do Sul de Minas Gerais, Varginha. Disponível em: http://repositorio.unis.edu.br/bitstream/prefix/601/1/TCC% 20-%20Monografia%20-%20KANBAN.pdf. Acesso em: 05 Dez 2020.

PRESNER, D. H. & JUNIOR, E. L. D. S. (2014). *Estudo Sobre Metodologias Ágeis de Desenvolvimento Aplicando a Metodologia Extreme Programming em uma Aplicação Web*. 61 f. Trabalho de Conclusão de Curso (Tecnólogo em Análise e Desenvolvimento de Sistemas) - Universidade Tecnológica Federal do Paraná, Ponta Grossa. Disponível em: http: //repositorio.roca.utfpr.edu.br/jspui/bitstream/1/5979/1/PG_COADS_2014_1_06.pdf. Acesso em: 04 Dez 2020.

RIANI, A. M. (2006). *Estudo De Caso: O Lean Manufacturing Aplicado na Becton Dickinson*. 52 f. Monografia (Engenharia de Produção) - Universidade Federal de Juiz de Fora, Juiz de Fora. Disponível em: https://www.ufjf.br/engenhariadeproducao/files/2014/09/ 2006_3_Aline.pdf. Acesso em: 09 Dez 2020.

ROYCE, W. (1970). *Managing the development of large* software *systems: concepts and techniques*. IEEE Westcon, Los Angeles, CA.

RUBIN, K. (2012). *Essential Scrum: A practical guide do the most popular agile process*. Addison-Wesley Professional.

SABBAGH, R. (2018). *As (Verdadeiras) origens do Scrum*.https://knowledge21.com.br/blog/asverdadeiras-origens-do-scrum/. Acesso em: 20 Out 2020.

SALGADO, E., MELLO, C., SILVA, C., OLIVEIRA, E. & ALMEIDA, D. (2009). *Análise da Aplicação do Mapeamento do Fluxo de Valor na Identificação de Desperdícios do Processo de Desenvolvimento de Produtos*. Gest. Prod., São Carlos, v. 16, n. 3, p. 344356, jul-set. Disponível em: https://www.scielo.br/pdf/gp/v16n3/v16n3a03.pdf. Acesso em: 09 Dez 2020.

SANTOS, J. V. P. D. (2015). *Uso do Kanban em um Processo de Gestão de Demandas de Manutenção de* Software *por Terceiros para um Órgão Público Federal Brasileiro*. 113 f.

Monografia (Bacharelado em Engenharia de Software) - Universidade de Brasília, Brasília. Disponível em: http://fga.unb.br/articles/0001/0082/TCC2_JadsVictorPaivadosSantos. pdf.. Acesso em: 04 Dez 2020.

SAVOINE, M., ROCHA, L., ROCHA, M. & SANTOS, C. (2009). *Análise de Gerenciamento de Projeto de* Software *Utilizando Metodologia Ágil XP e Scrum: Um Estudo de Caso Prático*. 10fl. (Artigo, Curso de Sistema de Informação) - Inst. Toc. Presidente Antônio Carlos, Tocantis. Disponível em: http://ulbra-to.br/encoinfo/wp-content/uploads/2020/03/An%5C%C3%5C%A1lise-de-Gerenciamento-de-Projeto-de-Software-Utilizando-Metodologia-%5C%C3%5C%81gil-XP-e-*Scrum*-Um-Estudo-de-CasoPr%5C%C3%5C%A1tico.pdf. Acessado em: 27 Set 2020.

SCHWABER, K. (2004). *Agile Project Management with Scrum*. Microsoft Press. Washington. Disponível em: https://poetiosity.files.wordpress.com/2011/04/agile_project_ management_with_scrum.pdf%20.. Acesso em: 05 Dez 2020.

SCHWABER, K. & SUTHERLAND, J. (2017). *Scrum Guide*. Disponível em: https://www. scrumguides.org/. Acesso em: 01 Set 2020.

SIMOYAMA, F. D. O., BUENO, R. L. P. & BATTISTI, M. C. G. (2016). *Adaptação e implantação da metodologia Scrum para projetos ágeis numa Autarquia Federal*. Revista Gestão & Tecnologia, Pedro Leopoldo, v. 16, n. 2, p. 260-276, maio./ago. Disponível em: http://revistagt.fpl.emnuvens.com.br/get/article/download/937/674. Acesso em: 05 Dez 2020.

SIQUEIRA, H. (2007). *Mapeamento das práticas de Scrum nas áreas de processo do CMMI e uma proposta para sua aderência*. Trabalho de Graduação (Ciência da Computação) Universidade Federal de Pernambuco, Pernambuco.

SOARES, B. D. (2019). *Diagnóstico de Métodos Ágeis para Empresas do Segmento de Tecnologia da Informação e Comunicação*. 57 f. Trabalho Conclusão do Curso (Tecnologias da Informação e Comunicação) - Universidade Federal de Santa Catarina, Araranguá. Disponível em: https://repositorio.ufsc.br/bitstream/handle/123456789/197476/TCC_BERNARDO_DALPIAZ_SOARES.pdf?sequence=1. Acesso em: 20 Dez 2020.

SOMMERVILLE, I. (2011). *Engenharia de* Software. 9 ed. São Paulo: Pearson Prentice Hall.
STANDISH GROUP Inc. (2001). *Extreme Chaos*. Disponível em: http://www.cin.ufpe.br/~gmp/docs/papers/extreme_chaos2001.pdf. Acesso em: 27/10/2020.

TAKEUCHI, H. & NONAKA, I. (1986). *The New New Product Development Game*. Harvard Business Review. Disponível em: https://hbr.org/1986/01/the-new-new-productdevelopment-game. Acesso em: 20 Out 2020.

TOMÁS, M. R. S. (2009). *Métodos ágeis: Características, Pontos Fortes e Fracos e Possibilidades de Aplicação*. 19 f. Monografia (Mestrado em Engenharia Informática) - Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa, Monte de Caparica. Disponível em: https://run.unl.pt/bitstream/10362/2003/1/WPSeries_09_2009Tomas.pdf. Acesso em : 03 Dez 2020.

UTIDA, H. K. (2012). *Metodologias Tradicionais E Metodologias Ágeis: Análise Comparativa entre Rational Unified Process e Extreme Programming*. 48 f. Monografia (Processamento de Dados) - Faculdade de Tecnologia do Estado de São Paulo, São Paulo. Disponivel em: http://www.fatecsp.br/dti/tcc/tcc00055.pdf. Acesso em: 03 Dez 2020.

VELOSO, C. E. F. (2006). *Uma proposta de Aplicação do Kanban no Controle de Estoque de uma Empresa Comercial de Pequeno Porte*. 54 f. Monografia (Engenharia de Produção) - Universidade Federal de Juiz de Fora, Juiz de Fora. Disponível em: https://www. ufjf.br/engenhariadeproducao/files/2014/09/2006_1_Carlos-Eduardo-Fernandes.pdf. Acesso em: 10 Dez 2020.

WAZLAWICK, R. S. (2013). *Engenharia de* Software*: conceitos e práticas*. Elsevier.

WOMACK, J., BYRNE, A., FIUME, O., KAPLAN, G. & TOUSSAINT, J. (2005). *Going Leanin Healthcare*. Innovation Series, Institute for Healthcare Improvement. Disponível em: http://www.ihi.org/resources/Pages/IHIWhitePapers/Going*Lean*inHealthCare.aspx. Acesso em: 15 Dez 2020.

WOMACK, J. & JONES, D. (1996). *Lean Thinking: Banish Waste and Create Wealth in Your Corporation*. 2ª ed. Free Press.

WOMACK, J., ROOS, D. & JONES, D. (1990). *A Máquina Que Mudou o Mundo*. 11. ed. Brasil: Elsevier. 342 p