UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

WELDEMIO JOSÉ DE OLIVEIRA DA SILVA CAVALCANTE

# UMA ANÁLISE DE FERRAMENTAS PARA AUXILIAR O GERENCIAMENTO ÁGIL DE PROJETOS DE SOFTWARE

DOIS VIZINHOS

2021

WELDEMIO JOSÉ DE OLIVEIRA DA SILVA CAVALCANTE

# UMA ANÁLISE DE FERRAMENTAS PARA AUXILIAR O GERENCIAMENTO ÁGIL DE PROJETOS DE SOFTWARE

# AN ANALYSIS OF TOOLS TO ASSIST THE AGILE MANAGEMENT OF SOFTWARE PROJECTS

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do título de Bacharel em Engenharia de Software da Universidade Tecnológica Federal do Paraná (UTFPR).

Orientador: Profa. Dra. Alinne Cristinne Corrêa Souza

**DOIS VIZINHOS**

**2021**

Ministério da Educação
**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ**
COORD. DO CURSO DE ENG. DE SOFTWARE-DV

**TERMO DE APROVAÇÃO**

## TRABALHO DE CONCLUSÃO DE CURSO - TCC

## UMA ANÁLISE DE FERRAMENTAS PARA AUXILIAR O GERENCIAMENTO ÁGIL DE PROJETOS

Por

Weldemio José de Oliveira da Silva Cavalcante

Monografia apresentada às 10horas 30min. do dia 13 de dezembro de 2021 como requisito parcial, para conclusão do Curso de Bacharelado em Engenharia de Software da Universidade Tecnológica Federal do Paraná, Câmpus Dois Vizinhos. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação e conferidas, bem como achadas conforme, as alterações indicadas pela Banca Examinadora, o trabalho de conclusão de curso foi considerado APROVADO.

Banca examinadora:

| | |
|---|---|
| Prof. Dr Gustavo Jansen de Souza Santos | Membro |
| Prof. Dr. Ricardo Ferreira Vilela | Membro |
| Profa. Dra. Alinne Cristinne Corrêa Souza | Orientadora |
| Prof. Dr. Francisco Carlos Monteiro Souza | Professor responsável TCCII |

## AGRADECIMENTOS

Agradeço primeiramente a Deus, por não desistir de mim, pois sem suas palavras não teria chegado até aqui.

Meus Pais, Ivan e Marines, por quebrarem o ciclo e me proporcionar uma educação de qualidade e dar esse grande passo.

Minha Esposa Bruna, a me apoiar, ajudar e compartilhar este momento.

Minha irmã Queren e meu Cunhado Leonardo, por me acolherem no inicio e fim dessa jornada, que sem eles, muita das vezes não teriam progredido.

Meus amigos, que me ajudaram diretamente e indiretamente, em provas, trabalhos, Marcos, Natália, Mateus, Raul, tantos que me ajudaram e que já seguiram seus caminhos. Dentre tantos minha orientadora, Alinne, que além de me ajudar, é também minha amiga, a qual nunca desistiu de mim, apesar de muitas e das mais diversas dificuldades nunca desistiu de mim, por isso tem minha gratidão.

## RESUMO

Em decorrência de grandes necessidades do mercado de trabalho, é notável a importância em conhecer ferramentas que auxiliem na melhoria de projetos de Software. Neste contexto, ferramentas para auxiliar o Gerenciamento Ágil de Projetos, vêm sendo debatidas a fim de transformar, auxiliar e aperfeiçoar os projetos para torná-los mais eficientes. Assim, este estudo tem como objetivo realizar uma análise comparativa entre ferramentas *opensources*, *freemium* e pagas com teste grátis para o Gerenciamento Ágil de Projetos. Nesta pesquisa foram analisadas 22 ferramentas por meio de 34 diretrizes definidas a partir dos trabalhos relacionados identificados, bem como a documentação e a avaliação do uso das ferramentas. Os resultados alcançados indicam que as ferramentas *Asana* e *Airtable* contemplaram mais de 88% das diretrizes. Portanto, este estudo apresentou diferentes características das ferramentas relacionadas à integração, escopo, cronograma, qualidade, riscos, comunicação, geração de artefatos e gerenciamento, a fim de auxiliar os usuários a escolherem a ferramenta que melhor atende suas necessidades.



**Palavras-chave:** Projeto de software, Gerenciamento de Projeto Ágil, Ferramentas

## ABSTRACT

Due to the significant needs of the job market, the importance of knowing tools that help the improvement of software projects is remarkable. In this context, tools to assist Agile Project Management have been investigated to transform, aid, and improve projects to make them more efficient. Thus, this study aims to carry out a comparative analysis between *opensources*, *freemium* and paid with free trial tools for Agile Project Management. In this research, 22 tools were analyzed through 34 guidelines defined from the related works identified, the documentation, and the evaluation of the use of tools.The results achieved indicate that the *Asana* and *Airtable* tools covered more than 88% of the guidelines. Therefore, this study presented different characteristics of the tools related to integration, scope, schedule, quality, risks, communication, generation of artifacts and management, to help users choose the tool that best meets their needs.



**Keywords:** Software project, Agile Management Project, Tools

## LISTA DE FIGURAS

## LISTA DE TABELAS

**SUMÁRIO**

# 1 INTRODUÇÃO

Com os avanços tecnológicos, independente do ramo de atuação no qual as organizações estão inseridas, o mercado está se tornando cada vez mais competitivo e complexo. Essa competitividade está exigindo mudanças rápidas e fazendo com que novas estratégias sejam desenvolvidas visando a conclusão dos projetos e, consequentemente, a melhoria da qualidade dos produtos e serviços.

O sucesso de um projeto está diretamente relacionado à sua conclusão considerando as restrições de escopo, cronograma, custo, qualidade, recursos, riscos, bem como a aprovação das decisões e produtos entregues pelas partes interessadas, também conhecidos como stakeholders (PMI, 2017).

No contexto de projetos de software, de acordo com uma pesquisa publicada em 2018 pelo *Standish Group*, apenas 28% dos projetos são considerados como concluídos com sucesso, 20% são tidos como fracassados e 52% como modificados em relação aos objetivos para os quais foram propostos (INTERNATIONAL, 2018). Assim, a forma como o projeto de software é gerenciado pode representar o seu sucesso ou fracasso, devido aos recursos humanos, financeiros, de tempo e todos os outros fatores que uma equipe de projeto possa estar suscetível.

Neste cenário, é notável que para determinados tipos de projetos de software, a metodologia a ser utilizada para o gerenciamento dos mesmos é um fator fundamental a ser decidido antes de iniciar um projeto, uma vez que pode determinar se os objetivos do mesmo serão alcançados como planejado ou não. Por este motivo é essencial que exista um bom gerenciamento dos projetos e que seja realizada a seleção da metodologia adequada, visando alcançar os requisitos definidos e as expectativas dos *stakeholders*.

Para organizar o gerenciamento de projetos de software com mais qualidade, muitas metodologias surgiram ao longo do tempo, como por exemplo as metodologias tradicionais. Essas metodologias possuem uma abordagem focada na documentação e que planejam especificar completamente os requisitos e, em seguida, projetar, construir e

testar o sistema, logo não estando adaptadas às constantes mudanças que ocorrem durante o desenvolvimento do software (SOMMERVILLE, 2016). Neste contexto, a metodologia tradicional tem sido cada vez menos utilizada em projetos devido a ser inflexível quanto à tomada de decisões em situações críticas; à burocracia, à quantidade de documentação; e à lentidão (PRIKLADNICKI; WILLI; MILANI, 2014).

Em contraposição às metodologias tradicionais, surgiu a metodologia de Manifesto Ágil. Essa metodologia visa o atendimento das expectativas e das necessidades do cliente, a entrega rápida de valor, o reconhecimento da capacidade dos indivíduos e, principalmente, a adaptação a ambientes de negócio (COHN; FORD, 2003).

Como exemplos das principais metodologias ágeis encontradas na literatura, podem ser citadas *Extreme Programming (XP), Scrum, Lean Software Development (LSD), Feature Driven Development (FDD), Adaptative Software Development (ASD), Dinamic Software Development Model (DSDM) e Crystal Methods.* Dentre as variações e adaptações das metodologias, o conceito das metodologias ágeis de desenvolvimento de software foi expandido, dando origem a um novo enfoque de gerenciamento de projetos conhecido como Gerenciamento Ágil de Projetos (GAP).

O GAP surgiu da necessidade de uma abordagem mais flexível para simplificar o desenvolvimento de software, de uma forma que fosse possível obter mais serviços e produtos, nos quais o desenvolvimento é desempenhado de acordo com a necessidade e mudanças frequentes no mercado. O GAP consiste em uma abordagem desenvolvida a partir de um conjunto de princípios e valores visando tornar o processo de gerenciamento de projetos simples, flexível e interativo (AMARAL, 2017).

No gerenciamento de projetos, é necessário que as necessidades e expectativas dos clientes sejam atendidas com excelência. Neste contexto, o GAP no desenvolvimento de software diminui os custos, oferece maior garantia de qualidade e produtividade e, maior satisfação do negócio (MISHRA; MISHRA, 2013). Para auxiliar esse gerenciamento, as ferramentas passaram a ser cada vez mais utilizadas pelas organizações, uma vez que buscam estabelecer práticas uniformes para gerenciar seus projetos, principalmente os maiores e mais complexos.

Com o objetivo de apoiar o GAP, foram surgindo diversas ferramentas como Jira[1], Asana[2], Pivotal Tracker[3] a fim de auxiliar no gerenciamento das atividades e

[1]https://www.atlassian.com/br/software/jira
[2]https://asana.com/pt
[3]https://www.pivotaltracker.com

gerar mais confiança nas entregas, melhorar organização, mais melhorias alcançadas, e consequentemente, usuários/clientes satisfeitos.

No entanto, apesar da diversidade de ferramentas, algumas podem não satisfazer as reais necessidades, podendo reduzir ou até mesmo impedir o desenvolvimento dos projetos (AZIZYAN, 2011); (BAJWA; KAUR, 2017); (ÖZKAN; MISHRA, 2019). É notório que não existe uma ferramenta completa e universal para as necessidades de qualquer projeto, uma vez que essas ferramentas tendem auxiliar as organizações e equipes de acordo com as suas reais necessidades e em projetos específicos. Neste contexto, um dos grandes desafios enfrentados tanto no âmbito acadêmico quanto na indústria é identificar qual ferramenta pode auxiliar o GAP, uma vez que a escolha da ferramenta depende totalmente das características da ferramenta e da demanda e necessidades dos usuários. Diante deste cenário, a questão de pesquisa que foi investigada é: Quais ferramentas são mais adequadas para apoiar o gerenciamento ágil de projetos?.

## 1.1 Objetivos

O presente trabalho apresenta uma análise de 22 ferramentas *opensources*, *freemiums* e pagas com teste grátis visando auxiliar o processo de seleção de ferramentas para apoiar o GAP. Para a condução da análise das ferramentas selecionadas foram definidas 34 diretrizes a fim de identificar suas similaridades, diferenças e características.

Para alcançar o objetivo geral os seguintes objetivos específicos foram delineados:

- selecionar um conjunto de ferramentas existentes para o gerenciamento ágil de projetos;
- definir um conjunto de diretrizes para avaliar as ferramentas selecionadas;
- realizar uma análise comparativa das ferramentas utilizando as diretrizes definidas.

## 1.2 Estrutura da Monografia

Esta monografia está organizada em quatro capítulos. No Capítulo 2, são definidos e apresentados os principais conceitos a respeito do gerenciamento de projetos, gerenciamento ágil de projetos e metodologias ágeis, bem como a discussão dos trabalhos relacionados.

No Capítulo 3, é apresentado o processo realizado para análise das ferramentas para auxiliar o gerenciamento ágil de projeto. No Capítulo 4 são apresentados e discutidos os resultados alcançados a partir da análise das 22 ferramentas. Finalmente, no Capítulo 5 são apresentados os resultados esperados, bem como as principais contribuições deste trabalho.

# 2 ASPECTOS CONCEITUAIS

Ao longo deste Capítulo, são apresentados os aspectos conceituais a fim de introduzir e contextualizar o assunto deste trabalho. Este capítulo está organizado da seguinte forma: Na Seção 2.1 são apresentados os conceitos e definições referentes ao Gerenciamento de projetos. Na Seção 2.2 é apresentado o GAP, bem como sua importância e princípios. Na Seção 2.3 são apresentadas as principais metodologias ágeis que podem auxiliar o gerenciamento ágil de projetos de software. Na Seção 2.4 são apresentados e discutidos os trabalhos relacionados. Por fim, na Seção 2.5 são apresentadas as considerações finais deste Capítulo.

## 2.1 Gerenciamento de Projetos

A definição de projeto consiste no esforço temporário proposto para criar um produto, serviço ou resultado único. Em outras palavras, a criação do projeto indica que seu início e fim são definidos (PMI, 2017). Um projeto termina quando é alcançado o seu objetivo ou a proposta inicial, ou pode ser encerrado sem estar finalizado, ou quando a necessidade do projeto deixa de existir. Neste contexto, a decisão de encerrar o projeto requer a aprovação de uma autoridade apropriada.

Segundo CARVALHO e JR. (2015), projeto é uma organização de pessoas cuja finalidade é alcançar um objetivo específico. O projeto envolve muitos fatores, como gastos, ações e empreendimento únicos com riscos elevados e possuindo um prazo para encerrar. Neste contexto, um projeto é desenvolvido a partir de uma ideia, progredindo para um plano, que, por sua vez é executado e concluído. Cada fase do projeto é caracterizada pela entrega ou finalização de um determinado trabalho. O projeto de maneira genérica pode ser dividido em cinco grandes etapas: iniciação, planejamento, execução, monitoramento e controle e, encerramento do projeto (PMI, 2017). Portanto, cada fase do projeto precisa ser gerenciada adequadamente para alcançar os objetivos definidos.

Assim, o gerenciamento de projetos consiste na aplicação de conhecimento,

habilidades, ferramentas e técnicas às atividades do projeto para atender aos seus requisitos (PMI, 2017). Para KERZNER (2015), o gerenciamento de projetos pode ser definido como planejamento, programação e controle de uma série de tarefas, com uma forma de alcançar os objetivos com êxito. Desta forma, para que as organizações executem projetos de forma eficaz, é necessária uma metodologia eficiente que guie o gerente e sua equipe ao longo dos projetos.

Independentemente do tipo de projeto, este deve ser bem gerenciado desde a concepção da ideia até a venda do produto, com os custos e os prazos dentro do previsto, com objetivo de minimizar os riscos para sua conclusão com sucesso. Neste contexto, foi crescendo a necessidade de diferentes abordagens que sejam adaptáveis e auxiliem o gerenciamento de projetos às diferentes situações. Segundo Leal (2008), projeto de software possui diversas particularidades que o tornam diferente dos demais tipos de projeto, tais como: complexibilidade e flexibilidade. Tendo em vista que o gerenciamento de projeto de software possui essas particularidades, surgiram as metodologias ágeis uma vez que são adaptativas, flexíveis, iterativas e orientadas a pessoas (MAGALHÃES; ROUILLER; VASCONCELOS, 2005).

## 2.2 Gerenciamento Ágil de Projetos

O GAP surgiu da necessidade de um gerenciamento de projetos mais flexível, interativo e simples. Desta forma, é possível obter produtos e/ou serviços melhores, respondendo sempre às frequentes mudanças que o possam ser propostas e alterações no mercado. Neste contexto, o GAP é um conjunto de princípios, valores e práticas que auxilia a equipe de projetos a entregar produtos ou serviços de valor em ambientes desafiadores (HIGHSMITH, 2004). Assim, o GAP pode ser visto como uma nova plataforma de projetos, que é aplicável em locais onde são técnicas tradicionais e com processos definidos, não são ideais (CHIN, 2004).

É importante destacar que o GAP tem sido utilizado em diversas áreas para melhoria de gerenciamento de projetos, pois uma vez inserido em um ambiente de grandes incertezas e desafios, o GAP procura ajudar as empresas de uma forma rápida nas mudanças propostas, envolvendo sempre os *stakeholders*. No GAP, uma vantagem que é importante destacar é o tempo em relação a todo o processo de produção de um software, produto e/ou serviço. Essa vantagem é notória, uma vez que o GAP buscar auxiliar nas rápidas mudanças, envolvendo o cliente nas propostas do projeto, permitindo sempre uma visão do que está sendo feito pela equipe e proporcionando uma evolução iterativa do projeto.

Em geral, o GAP é uma forma simples, flexível e interativa, visando sempre os melhores resultados, menor esforço de gerenciamento e maiores níveis de inovação e agregação de valor ao cliente (AMARAL, 2017). Neste contexto, o GAP tem como base, princípios que representam seus valores e objetivos. Esses princípios foram sintetizados por diferentes autores para que sejam utilizados em diferentes tipos de gerenciamento de projetos e não apenas de software, mas em ambientes desafiadores. Neste contexto, AUGUSTNE (2005), Leac (2005), HIGHSMITH (2004), CHIN (2004) e BOEHM (2002), sintetizaram nove princípios que podem caracterizar o GAP, conforme pode ser visto na Tabela 1.

**Tabela 1 - Princípios do GAP**

| **Princípios** | **Descrição** |
|---|---|
| 1. Simplicidade | Promove a interação no desenvolvimento do projeto, proporcionando sempre a inovação. É proposta a substituição de planos detalhados pelas técnicas simples e visuais. |
| 2. Flexibilidade | Está relacionada à capacidade de absorver diversas mudanças, adaptando projetos e pessoas aos desafios promovidos em ambientes de negócios. |
| 3. Busca por excelência técnica | Visa assegurar a qualidade na realização das tarefas, a fim de reduzir esforços na verificação e correção; |
| 4. Agregar valor para o cliente e a equipe de projetos | Visa o desenvolvimento colaborativo, que envolve cliente e equipe de projetos, a fim de proporcionar uma maior eficiência e transparência na entrega final dos produtos. |
| 5. Iterações e entregas parciais | O desenvolvimento de projetos de maneira iterativa está relacionado à agilidade e adaptabilidade, pois proporciona visões parciais do produto em intervalos curtos de tempo, permitindo que o cliente tenha uma visão clara do que está sendo desenvolvido e possa colaborar com *feedbacks* precisos. |
| 6. Autogestão e auto-organização | As equipes devem ser auto-gerenciáveis e colaborativas na execução das atividades, participando ativamente da ativação, execução, planejamento e controle das atividades. Neste âmbito, o gerente tem a função de facilitar a visão do produto e guiar a equipe, transformando em resultados que agreguem valor no que está sendo realizado. |
| 7. Tomada de decisão participativa | Incentivo à interação e participação da equipe nas decisões, ou seja, seus membros devem ter a liberdade de participar das decisões de forma proativa. |
| 8. Inovação e criatividade | Incentivo à autogestão e auto-organização, é aberto um espaço para encorajar os membros da equipe a criar um ambiente favorável à inovação e criatividade. |
| 9. Comunicação | A comunicação é fundamental para que todos os princípios sejam eficazes. Essa comunicação deve ser pessoal, aberta e efetiva entre os envolvidos, uma vez que encoraja o *feedback*, constrói confiança e transforma palavras em ações. |

Portanto, o GAP surgiu como alternativa inovadora visando permitir a participação de todos os envolvidos com o projeto, a participação ativa do cliente, entregas frequentes com ciclos rápidos e perfil facilitador do gerente.

## 2.3 Metodologias Ágeis

Ao longo dos últimos anos, diversas metodologias ágeis foram criadas para simplificar o processo de gerenciamento de projetos. Segundo Amaral (2017), as metodologias podem ser divididas em duas vertentes: *(i)* aplicáveis a qualquer contexto de projeto; e *(ii)* aplicáveis ao desenvolvimento de software, sendo a segunda o foco deste trabalho.

Essas metodologias surgiram a partir de 2001, quando um grupo de 17 especialistas e representantes reuniram-se para discutir o padrão de gerenciamento e desenvolvimento de projetos entre as técnicas e metodologias existentes. O resultado desse encontro foi a criação do Manifesto para Desenvolvimento Ágil de Software, que estabeleceu um *framework* comum para processos ágeis, valorizando os seguintes itens: foco nos indivíduos e suas interações, software em funcionamento, colaboração com o cliente e respostas a mudanças (MANIFESTO, 2001). As subseções a seguir apresentam um resumo sobre as principais metodologias ágeis, de acordo com Agile Guide (2017).

### 2.3.1 Scrum

O Scrum, desenvolvido por Ken Schwaber e Jeff Sutherland, é um *framework* incremental e iterativo, orientado para a gerência dos projetos de software. O desenvolvimento é realizado por meio de iterações chamadas de *Sprints*, cada uma podendo ter a duração de duas a quatro semanas, denominado de *time-box*. O *framework* Scrum é baseado em um conjunto de papéis, eventos, artefatos e regras, cada qual com um propósito e sua importância para o sucesso do Scrum (SCHWABER; SUTHERLAND, 2017), conforme é apresentado na Tabela 2.

**Tabela 2 - Papéis, eventos e artefatos do Scrum**

| **Papéis** | **Eventos** | **Artefatos** |
|---|---|---|
| *Product Owner (PO)* | *Sprint* | *Product backlog* |
| *Scrum Master (SM)* | *Sprint planning* | *Sprint backlog* |
| Time | *Daily scrum* | Incrementos |
| | *Sprint review* | |
| | *Sprint retrospective* | |

A Figura 1 apresenta o fluxo de gerenciamento de projetos por meio do Scrum. A partir da visão do negócio, é criado uma lista de requisitos do projeto, chamada de *Product backlog*. Esses requisitos são priorizados pelo dono do produto conhecido como *PO*. Antes iniciar o Sprint, ocorre a reunião de *Sprint planning*, na qual o time se reúne com PO para negociar quais tarefas do *Product backlog* que serão desenvolvidas na *sprint*, este conjunto é chamado de *Sprint backlog*. Durante as Sprints, encontros diários conhecidos como *Daily scrum* de curta duração (10 a 15 minutos) são realizados para acompanhar o andamento das atividades e se há impedimentos. Ao final do *Sprint*, os resultados são apresentados ao PO e aos *stakeholders* interessados, e são realizadas reuniões de revisão e retrospectiva conhecidas como *Sprint review* e de *Sprint retrospective* para aprender e melhorar para os próximos *Sprints*.

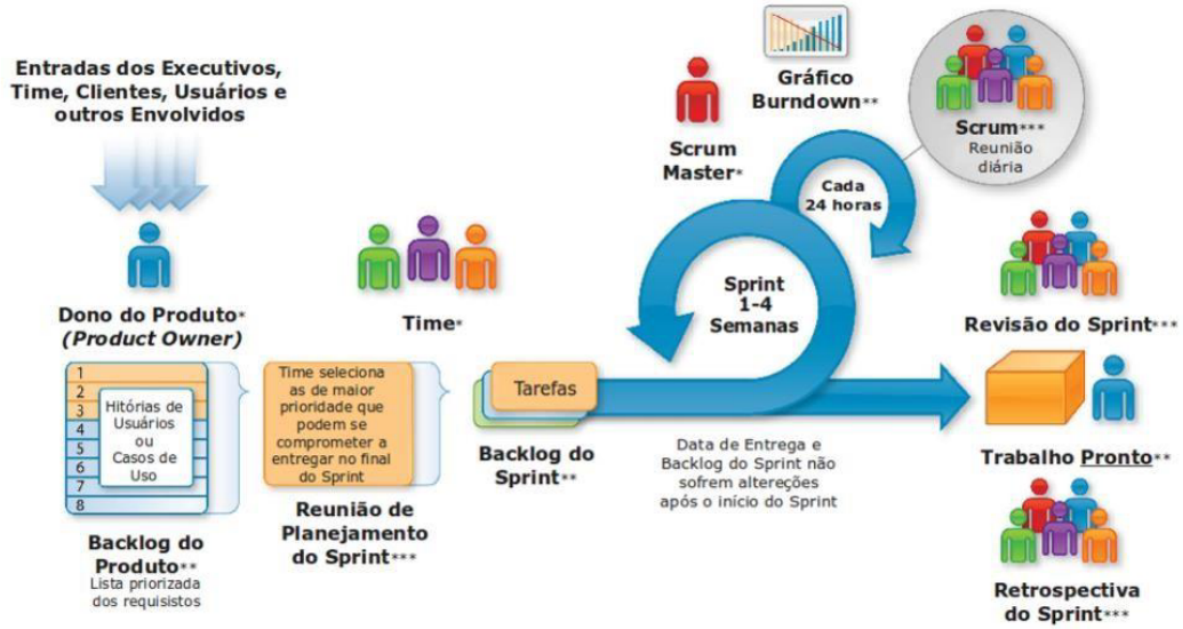


**Figura 1 - Processo de desenvolvimento do Scrum.**
**Fonte: Softhouse (2014).**

### 2.3.2 EXTREME PROGRAMMING

Criado em 1997, por Kent Beck, o *eXtreme Programming (XP)* é um método de desenvolvimento de software baseado em ciclos frequentes e na aplicação de uma determinada prática na sua forma mais pura e simples durante todo o projeto. O XP em sua ênfase em um rápido desenvolvimento do projeto e visa garantir a satisfação do cliente, além de favorecer o cumprimento do que foi proposto.

Segundo BECK (1999), os seguintes valores são a base do XP, sendo eles: *(i)* simplicidade, opta por processos mais simples e que fazem somente o necessário; *(ii)* comunicação, manter uma comunicação fluida por meio de práticas que unam a equipe e o cliente; *(iii) feedback*, consiste nos relatos das alterações necessárias e na busca de uma integração rápida e contínua; *(iv)* respeito, por meio da valorização e entendimento dos diversos interessados no projeto e sua experiências; e *(v)* coragem, promovida pelos quatro valores e foca nas aplicações e alterações.

O XP possui uma abordagem dinâmica e adaptativa que se adéqua conforme a necessidade de cada projeto. Segundo Pressman (2016), o XP propõem quatro atividades metodológicas principais: planejamento, projeto, codificação e testes, conforme podem ser vistas na Figura 2.


**Figura 2 - Processo de desenvolvimento do XP.**
**Fonte: Pressman (2016).**

A primeira atividade (Planejamento) consiste em entender o negócio para que sejam criados os requisitos por meio de histórias do usuário. Uma vez que as histórias são escritas, o valor e o custo são atribuídos às mesmas para que possam ser priorizadas. A segunda atividade (Projeto) ocorre tanto antes como após a codificação, ou seja, significa que o design é atualizado continuamente a medida que se constrói o software. Nesta atividade são utilizados os cartões CRC (Classe - Responsabilidade - Colaborador) na qual são identificadas e organizadas as classes com um escopo, preferencialmente, orientadas a objeto relevantes para a versão. Vale destacar que, os cartões CRC tem três atributos

principais que são a classe a responsabilidade e o colaborador. O nome da classe descreve o nome do objeto no contexto geral do ambiente, as responsabilidades buscam identificar os problemas a serem resolvidos, os colaboradores enviam mensagens ao objeto para atenderem suas responsabilidades (PRESSMAN, 2016).

A terceira atividade (Codificação) consiste no desenvolvimento de software com testes individuais das histórias incluídas na versão atual. Para isso, são utilizadas boas práticas de programação em par e realizada a integração continua do código com o resto da equipe. Por fim, na quarta atividade (Teste), após os testes individuais citados na etapa de codificação, deverão ser realizados os testes de aceitação especificadas pelo cliente. É importante destacar que todos os testes são oriundos das histórias dos usuários criadas e implementadas na versão.

### 2.3.3 FEATURE-DRIVEN DEVELOPMENT

O *Feature-Driven Development* (FDD) criado por Peter Coad e Jeff De Luca em 1999, é um método incremental e orientado a funcionalidades que visa atender às necessidades específicas de um grande projeto de desenvolvimento de software (GUIDE, 2017). O FDD se baseia em processos bem definidos e que podem ser repetidos. Sua abordagem se concentra nas fases de projeto e construção, com maior ênfase na modelagem, em um ciclo de vida iterativo e também em atividades de gerenciamento de projetos (UDO; KOPPENSTEINER, 2003).

De acordo com Agile (GUIDE, 2017), o processo do FDD é composto por cinco atividades: *(i)* desenvolvimento do modelo abrangente; *(ii)* construção da lista de funcionalidades; *(iii)* planejar por funcionalidade; *(iv)* detalhe por funcionalidade; e *(v)* construção por funcionalidade, conforme é apresentado na Figura 3.

A primeira atividade (desenvolvimento do modelo abrangente) visa desenvolver um modelo básico, a fim de gerar um arcabouço para sua construção nos outros processos. A segunda atividade (construção da lista de funcionalidades) consiste na listagem dos requisitos do sistema. A terceira etapa (planejar por funcionalidade) visa estimar e priorizar as funcionalidades, além de atribuí-las para seus respectivos desenvolvedores. Na quarta atividade (detalhe por funcionalidade), são atribuídas as atividades a serem realizadas em cada funcionalidade, e são projetados modelos e diagramas de apoio. Por fim, a quinta e última atividade (detalhe por funcionalidade) consiste na geração de código fonte para cada funcionalidade, agregando valor ao produto e ao cliente (UDO; KOPPENSTEINER, 2003).

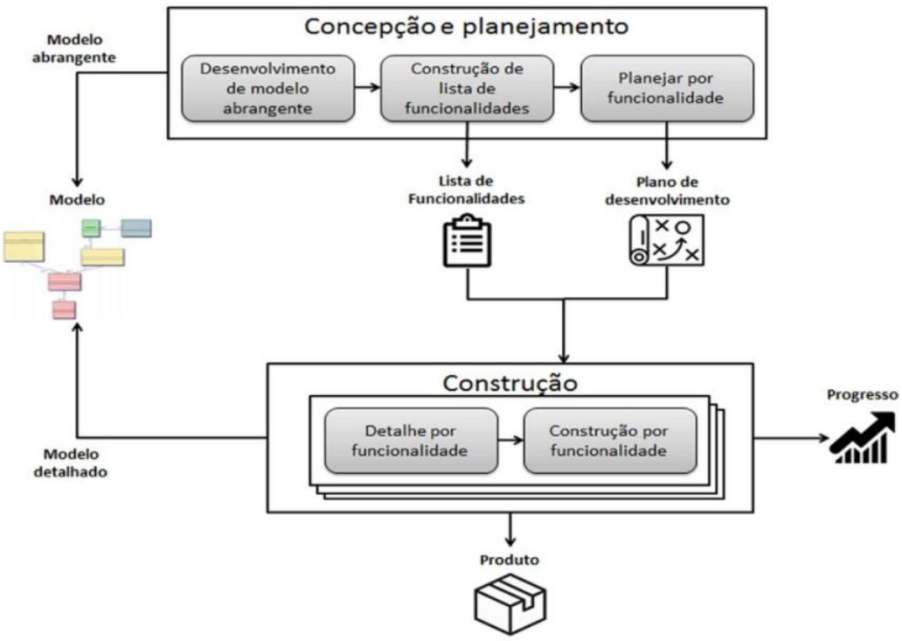


**Figura 3 - Processo de desenvolvimento do FDD.**
**Fonte: Silva e Jr (2018).**

### 2.3.4 CRYSTAL METHOD

Desenvolvido por Alistair Cockburn, possui uma série de métodos que relacionam o tamanho da equipe, a prioridade do projeto e complexidade do sistema (COCKBURN, 2004). Apesar de suas variações, o método Crystal foca a eficiência e habilidade da equipe como componente de garantia de qualidade do projeto. Além disso, o método foca as pessoas e não as atividades e artefatos. Segundo COCKBURN (2004), as principais características são:

- Entrega frequente de código utilizável pelo usuário;
- Melhoria contínua;
- Foco – trabalhar sobre as prioridades;
- Fácil acesso para usuários experientes;
- Testes automatizados, gerência de configuração e integração frequente.

O método Crystal destaca que cada projeto pode exigir um conjunto adaptado de políticas, práticas e processos, a fim de atender às características exclusivas de um determinado projeto. Um projeto gerenciado com o Crystal é bastante modularizado,

uma vez que é dividido em várias entregas, as entregas são divididas em interações, essas interações são compostas de reuniões de planejamento e reuniões diárias. As reuniões diárias são compostas de integrações de código e cada integração é composta de episódios e testes.

### 2.3.5 Dynamic Systems Development Method (DSDM)

DSDM foi criado na Inglaterra nos anos 90 para atender à necessidade de negócios rápidos. O DSDM é baseado no modelo de desenvolvimento de softwares chamado *"Rapid Application Development" (RAD)* que consiste em uma abordagem iterativa e incremental que enfatiza o envolvimento contínuo entre cliente e equipe. O DSDM é um método voltado para o desenvolvimento de software de forma que este não ultrapasse os limites de tempo e orçamento, mesmo assim, garantindo a qualidade. Este método é mais conhecido por sua ênfase na entrega orientada a restrições. A estrutura irá definir custo, qualidade e tempo no início e, em seguida, usar a priorização formalizada do escopo para atender a essas restrições Guide (2017).

Segundo Consortium (2014), o DSDM é baseado em oito princípios-chave, focados nas necessidades do cliente versus valor de negócio, tais como: *(i)* foco nas necessidades do negócio; *(ii)* entregar no prazo; *(iii)* colaboração; *(iv)* nunca comprometer a qualidade; *(v)* construir de forma incremental a partir de uma solida fundação; *(vi)* desenvolver iterativamente; *(vii)* comunicação clara e contínua; e *(viii)* demonstrar controle.

O ciclo de vida do DSDM é composto por quatro fases principais: Viabilidade, Fundações, Desenvolvimento Evolutivo e Implantação. Estas são precedidas pela Fase de Pré-Projeto e seguidas pela Fase de Pós-Projeto, totalizando seis fases, conforme é apresentado na Figura 4.

A fase de pré-projeto garante que apenas os projetos certos sejam iniciados e que eles sejam configurados corretamente, com base em um objetivo claramente definido. A fase de Viabilidade visa estabelecer se o projeto proposto é viável do ponto de vista técnico e se parece econômico do ponto de vista comercial. O esforço associado à viabilidade deve ser apenas o suficiente para decidir se é necessário investigação mais aprofundada ou se o projeto deve ser interrompido agora, por ser improvável que seja viável.

A fase de negócio tem como objetivo estabelecer uma compreensão fundamental, mas não detalhada, da lógica do negócio para o projeto, a solução potencial que será criada pelo projeto e como o desenvolvimento e a entrega da solução serão gerenciados. Portanto,

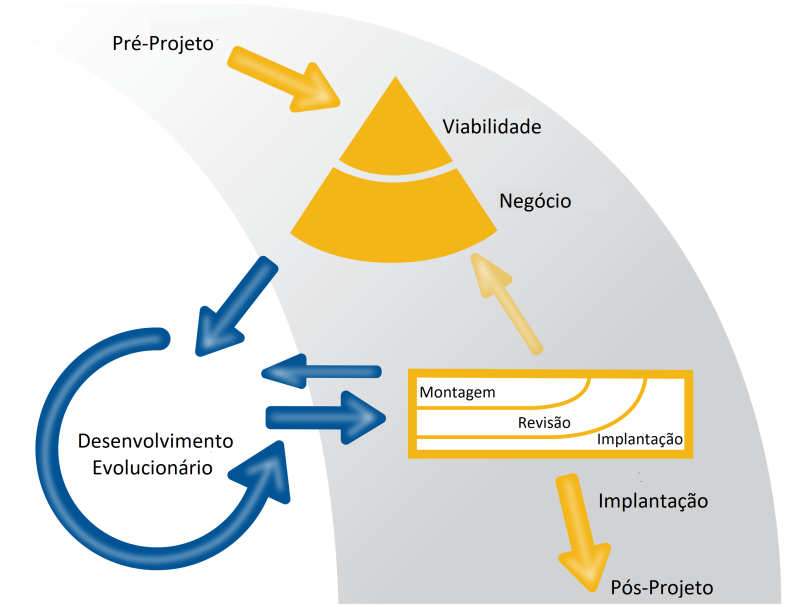


**Figura 4 - Ciclo de vida do DSDM.**
**Fonte: Adaptado de Consortium (2014).**

esta visa compreender o âmbito do trabalho, como será realizado, por quem, quando e onde.

Com base no que foi estabelecido para o projeto, o objetivo da fase de Desenvolvimento Evolucionário é desenvolver a solução. Nesta fase, as equipes de desenvolvimento devem aplicar práticas como desenvolvimento iterativo, *timeboxing* e priorização *MoSCoW*[1] e modelagem, para convergir ao longo do tempo em uma solução precisa que atenda às necessidades do negócio. Com a equipe de desenvolvimento trabalhando dentro de *timeboxes*, é possível criar incrementos de solução, explorando iterativamente os detalhes de baixo nível dos requisitos e testando continuamente conforme eles avançam.

O objetivo da fase de Implantação é colocar uma solução em evolução em uso operacional. Esta fase compreende três atividades principais: montagem, revisão e implantação. A atividade de montagem engloba o trabalho de "reunir" o que está para ser lançado. A atividade de revisão consiste na revisão final da solução antes de entrar em uso operacional para garantir que a versão proposta atenda os padrões apropriados e é completo o suficiente para ser viável. Após concedida a aprovação, a atividade de implantação visa colocar o que foi montado em uso operacional. Por fim, após a implantação final de um projeto, a fase Pós-projeto verifica se os benefícios comerciais esperados foram atendidos.

[1]É uma técnica de classificar e priorizar requisitos para inclusão em um sistema de informação (ORACLE, 2000).

## 2.4 Trabalhos Relacionados

O gerente de projetos, além de possuir habilidades e conhecimentos, deve também utilizar ferramentas que dão apoio às suas atividades. No entanto, identificar a melhor ferramenta de metodologia ágil para problemas específicos é um grande desafio para o desenvolvedor e o cliente, pois a base é a comunicação. Existem várias ferramentas de gerenciamento comerciais e de código aberto, que vão desde aplicativos sofisticados baseados na web até utilitários simples e especializados.

Nesta seção são apresentados oito trabalhos que conduziram uma análise e avaliação comparativa de ferramentas que fornecem suporte ao GAP. Os trabalhos foram identificados por meio de uma revisão da literatura narrativa, a qual foi realizada por meio da busca de trabalhos no *Google Schoolar*.

Özkan e Mishra (2019) investigaram ferramentas de gerenciamento de projeto ágeis, populares entre profissionais de software, e fizeram uma comparação dos seus recursos, com o objetivo de obter melhores resultados em sua utilização, e entender como isso afeta na comunicação da equipe, e alocação dos recursos. Foram selecionadas 16 ferramentas, sendo elas *open source* e pagas, tais como: Jira[2], ActiveCollab[3], Agilo for Scrum[4], SpiraTeam[5], Pivotal Tracker[6], Microsoft Visual Studio Team Services (VSTS) hoje conhecida como Azure DevOps[7], Icescrum[8], SprintGround[9], VersionOne[10], Taiga[11], Agilean[12], Wrike[13], Trello[14], Axosoft[15], Planbox[16] e Asana[17]. As ferramentas foram analisadas com base em 19 características: baseada em plataforma, baseada em web, online, baseada em nuvem, gráfico *burndown*, *agile boards*, marco históricos, gerenciamento de recursos, rastreamento de tempo, rastreamento de bug, tarefas, integração, relatórios, documentos, controle de versão, espaço de trabalho, papel do usuário, preço e versão gratuita. Os resultados indicaram que as ferramentas mais apreciadas são Jira, Trello e VersionOne. A ferramenta

[2]https://www.atlassian.com/br/software/jira
[3]https://activecollab.com/
[4]http://freshmeat.sourceforge.net/projects/agiloforscrum
[5]https://www.inflectra.com/SpiraTeam/
[6]https://www.pivotaltracker.com/
[7]https://azure.microsoft.com/
[8]https://www.icescrum.com/
[9]https://www.sprintground.com/
[10]https://www.collab.net/
[11]https://www.taiga.io/
[12]https://www.agilean.com.br/
[13]https://www.wrike.com/
[14]https://trello.com
[15]https://www.axosoft.com/
[16]https://www.planbox.com/
[17]https://asana.com

Jira é popular pois suporta 15 das 19 características. Por outro lado, as ferramentas Taiga, Axosoft, Agielan, Planbox, foram consideradas adequadas somente para projetos de *startups*.

No estudo de Bajwa e Kaur (2017) sobre ferramentas de GAP, a partir dos quais foi conduzida uma análise comparativa com base nas funcionalidades, características comuns e diferentes, e as vantagens e desvantagens de cada ferramenta. De acordo com o estudo foram identificadas 10 ferramentas: Trac[18], Rally[19], VersionOne, Jira, Bugzilla[20], Assembla, Pivolt Tracker, ActiveCollab, Agilo for Scrum e Mantis[21]. A análise das ferramentas foi realizada considerando 13 características/critérios: *(i) opensource*; *(ii)* criação e configuração de projetos; *(iii)* manipulação de múltiplos projetos; *(iv)* fluxo de trabalho personalizável; *(v)* rastreamento de tempo; *(vi)* integração com outros programas; *(vii)* integração com outros programas; *(viii)* aplicativos móveis; *(ix)* rastreamento de bug; *(x)* notificações de email; *(xi)* busca avançada de funcionalidades; *(xii) dashboards*; e *(xiii)* gráfico de Gantt. De acordo com os resultados, as ferramentas Jira, Bugzilla, Assembla e Mantis fornecem o mais alto nível de recursos para gerenciar projetos ágeis em comparação com outras ferramentas. No entanto, Bugzilla e Mantis são ferramentas de código aberto e as outras duas são ferramentas proprietárias.

O estudo desenvolvido por Mihalache (2017) realizou uma análise de quatro ferramentas comerciais de gerenciamento ágil de projetos (Jira, VersionOne, Rally e Azure DevOps) e propôs um modelo de classificação para a seleção de ferramentas ágil apropriada. Para realizar uma comparação eficaz, a autora analisou três recursos (relatórios e métricas ágeis, comunicação e avaliação do projeto) e definiu uma lista contendo oito critérios importantes que as ferramentas de gerenciamento ágil devem ter para satisfazer as equipes ágeis, tais como: *(i)* necessidade de negócio; *(ii)* custo; *(iii)* visualização do progresso; *(iv)* rastreamento de problema e tempo; *(v)* colaboração; *(vi)* estimativa; *(vii)* portfólio do projeto; e *(viii)* portal do cliente. Além das ferramentas comerciais, a autora realiza uma análise das planilhas (Google suite[22] e Microsoft Project[23]) e ferramentas simples (quadro branco/*post-it*) a fim de verificar se as equipes de desenvolvimento precisam usar ferramentas caras para produzir um bom software. De acordo com os resultados, a ferramenta Jira é apropriada para gerenciamento de projetos, rastreamento de defeitos e possui a disponibilidade de uma grande quantidade de *plugins*. Já o Google Docs

[18]http://www.trac.edgewall.org
[19]http://www.rallydev.com
[20]http://www.bugzilla.org/
[21]https://www.mantisbt.org/
[22]https://workspace.google.com
[23]https://www.microsoft.com/pt-br/microsoft-365/project/project-management-software

e o Microsoft Project são ferramentas eficazes para pequenas equipes com processos relativamente simples.

No estudo de Manole e Avramescu (2017) foi conduzida uma análise comparativa de cinco ferramentas mais utilizadas pelas empresas, tais como: Jira, Taiga, VersionOne, Assembla[24] e Asana. Para realizar essa análise comparativa, os autores definiram cinco critérios que são relevantes para o contexto ágil: *(i)* facilidades de integração - são fundamentais para uma boa colaboração, uma vez que o time e/ou cliente podem estar distribuídos geograficamente e o fluxo de informações a serem trocadas entre os envolvidos é grande; *(ii)* aplicativos móveis - disponibilidade em múltiplas plataformas oferecendo flexibilidade e, por outro lado, a possibilidade de integração com a nuvem para armazenamento de grandes volumes de dados e até trabalho colaborativo remoto; *(iii)* histórias do usuário - são curtas, simples e descrevem em linguagem natural a perspectiva de uma pessoa (usuário ou cliente) sobre uma funcionalidade do projeto; *(iv)* treinamento - visa auxiliar no aprendizado e compreensão de funcionalidades das ferramentas e como podem ser utilizadas para melhorar as atividades e as produtividades das empresas; *(v)* gerenciamento de tarefas - ajuda todos os envolvidos a saberem sobre o andamento do projeto e a equipe para organizar melhor seus recursos, bem como manter o custo esperado do projeto. De acordo com os critérios, a ferramenta com os melhores indicadores foi a Jira em todos os critérios, seguida pela Asana que possui gerenciamento de tarefas a partir das necessidades dos desenvolvedores.

Na pesquisa desenvolvida por Buturugã, Gogoi e Prodan (2016) foi realizada uma análise das ferramentas Jira e Tuleap[25] com equipes de desenvolvimento. O objetivo é avaliar se as ferramentas oferecem controle contínuo e permanente sobre o projeto em que a equipe de desenvolvimento está trabalhando. Para isso, foram considerados três critérios: *(i)* relatórios e métricas ágeis; *(ii)* comunicação (comunicar atualizações, *feedback*, compartilhar lista de tarefas e estórias de usuário); e *(iii)* avaliação do projeto. Com os resultados, foi possível perceber que a ferramenta Jira, pode ser utilizada em projetos de todos os portes, desde pequenosa grandes. Também permite as equipes adotar quaisquer práticas ágeis como: *ASD, Crystal Method, Scrum, Scrum ban*, ou metodologias mistas. Já a ferramenta Tuleap possui uma licença pública geral, o que a torna mais acessível aos usuários, porém não possui softwares de comunicação, o que poderia tornar o trabalho menos eficiente.

Azizyan (2011) realizou uma pesquisa pela empresa de telecomunicações Ericsson,

[24]https://www.assembla.com/home
[25]https://www.tuleap.org/

com o objetivo de analisar o uso e a necessidade das ferramentas ágeis. Os autores conduziram um *survey*, por meio do qual foram coletadas 121 respostas de 120 empresas em 35 países, em que os programadores listaram os aspectos mais e menos eficazes das ferramentas, e quais recursos adicionais poderiam ser utilizados para melhorar o trabalho. Para a análise das ferramentas no *survey* foram utilizados cinco critérios: *(i)* facilidade de uso; *(ii)* integração com outros programas; *(iii)* personalização - quão flexíveis as ferramentas são e quão fácil é adaptá-las às necessidades da empresa; *(iv)* disponibilidade de relatórios; e *(v)* preço. Os resultados demonstraram que entre os métodos ágeis mais utilizados entre as empresas, 54% delas usavam Scrum, 32% Scrum XP, 11% XP, 9% Kanban, 2,5% TDD, 2,5% Crystal, 2% DSDM e 2% FDD. O que indicou que mais de uma metodologia ágil era utilizada pelos profissionais. Já entre as ferramentas ágeis mais utilizadas estavam: 26% parede e papel físico, 23% de planilhas, 8% MS Project, 5% de ferramentas internas da empresa, 2% VersionOne, 5% Rally, 3% ScrumWorks, 3% Mingle, 2% XPlanner, 2% TFS, 4% Trac, 1% Assembla, 2% JIRA, e 12% de outras ferramentas.

No trabalho de Oliveira (OLIVEIRA, 2010) foram avaliadas algumas ferramentas com o objetivo, de descobrir quais seriam eficazes na gestão das organizações. Foi utilizado um padrão de qualidade dentro do gerenciamento de projetos dividido em nove áreas que são: gestão de integração do gerenciamento do projeto, gerenciamento do escopo, gerenciamento de tempo, gerenciamento de custos, gerenciamento de recursos humanos, gerenciamento de qualidade, gerenciamento de comunicação, gerenciamento de riscos, e gerenciamento de aquisição. A ferramentas gestoras de projetos analisadas foram: ClockingIT, OpenProj e Redmine, as quais eram *opensource*. A ferramenta que melhor atendeu todos os requisitos foi a Redmine com os nove critérios, a ClockingIT atendeu oito, e a OpenProj alcançou sete das áreas analisadas.

Na pesquisa de Lima (LIMA, 2021) a equipe de robótica Titãs da Robótica encontrou dificuldades no gerenciamento de projetos, e para facilitar essa gerência buscaram ferramentas que atendessem as demandas da equipe. Para tanto, foram feitos dois questionários. O primeiro foi estabelecido para a escolha das ferramentas. Elas deveriam ter versão gratuita, estar disponíveis em todas as plataformas, ter compartilhamento das informações entre usuários, e ter hierarquia na exibição das informações. Foram selecionadas dez ferramentas, entre elas: Asana, Bitrix 24, ClickUp, Confluence, Gantter, Open Project, Podio, Project Libre, Team Work, e Trello. O segundo questionário foi utilizado na escolha dos requisitos que as ferramentas selecionadas deveriam atender.

A Tabela 3 apresenta o resumo dos trabalhos relacionados discutidos anteriormente.

A primeira coluna lista os estudos relacionados, o número de ferramentas analisadas (segunda coluna) e os critérios/características utilizados para avaliar as ferramentas(terceira coluna).

**Tabela 3 - Resumo dos trabalhos relacionados.**

| Estudos | Ferramentas | Critérios/Características |
|---|---|---|
| Özkan e Mishra (2019) | 16 | 19 características: baseada em plataforma, baseada em web, online, baseada em nuvem, gráfico *burndown*, *agile boards*, marco históricos, gerenciamento de recursos, rastreamento de tempo, rastreamento de bug, tarefas, integração, relatórios, documentos, controle de versão, espaço de trabalho, papel do usuário, preço e versão gratuita. |
| Bajwa e Kaur (2017) | 10 | 8 características: *opensource*, criação e configuração de projetos, manipulação de múltiplos projetos, fluxo de trabalho personalizável, rastreamento de tempo, integração com outros programas, integração com outros programas, aplicativos móveis, rastreamento de bug, notificações de email, busca avançada de funcionalidades, *dashboards*, e gráfico de Gantt. |
| Mihalache (2017) | 4 comerciais, 2 planilhas e 1 simples | 8 critérios: necessidade de negócio, custo, visualização do progresso, rastreamento de problema e tempo, colaboração, estimativa, portfólio do projeto e portal do cliente. |
| Manole e Avramescu (2017) | 5 | 5 critérios: facilidades de integração, aplicativos móveis, histórias do usuário, treinamento e gerenciamento de tarefas. |
| Buturugã, Gogoi e Prodan (2016) | 2 | 3 critérios: relatórios e métricas ágeis, comunicação e avaliação do projeto. |
| Azizyan (2011) | - | 5 critérios: facilidade de uso, integração com outros programas, personalização, disponibilidade de relatórios e preço. |
| (LIMA, 2021) | 3 | 9 critérios: gestão de integração, gerenciamento do projeto, gerenciamento do escopo, gerenciamento de tempo, gerenciamento de custos, gerenciamento de recursos humanos, gerenciamento de qualidade, gerenciamento de comunicação, gerenciamento de riscos, e gerenciamento de aquisição. |
| (OLIVEIRA, 2010) | 10 | 6 critérios: número de usuários ilimitados, possibilita a troca de mensagens entre os usuários, possibilita a criação de tarefas, cronometra as tarefas, e definição de um coordenador. |

**Fonte: Autoria própria.**

## 2.5 Considerações Finais

Este Capítulo apresentou o referencial teórico sobre o gerenciamento de projetos, gerenciamento de projetos de software, gerenciamento ágil de projetos, metodologias ágeis de desenvolvimento de software, além dos principais trabalhos relacionados com a proposta deste projeto. Foram identificados diversos estudos comparativos de ferramentas que oferecem suporte ao gerenciamento ágil de projetos. No entanto, também foi possível identificar que embora muitas ferramentas tenham sido analisadas, apenas o estudo de Mihalache (2017) analisa planilhas e ferramentas simples.

É importante destacar também, que as mesmas características e/ou critérios foram utilizados por diferentes estudos para avaliar, muitas vezes, as mesmas ferramentas. Neste contexto, nenhum estudo indicou a ferramenta *opensource* ou paga mais adequadas, de acordo com os critérios, para que o gerente tenha conhecimento disso antes de aplicar a ferramenta que ele escolher. Portanto, justificando a importância desse estudo para a análise das ferramentas a ser apresentada no Capítulo 3.

# 3 ANÁLISE DAS FERRAMENTAS PARA AUXILIAR O GERENCIAMENTO ÁGIL DE PROJETOS DE SOFTWARE

A metodologia ágil surgiu mediante a melhoria necessária a partir da metodologia tradicional, de maneira que foram surgindo diversas questões, enfatizando que a evolução dessas metodologias era necessária. Em virtude disto, é sabido que a metodologia ágil é composta por diversos aspectos, sendo os mesmos aplicados de forma a agilizar e melhorar o processo, possibilitando uma melhor utilização de recursos para a equipe, melhor interação com cliente, e consequentemente gerando a redução dos gastos, a fim de tornar os processos mais favoráveis durante o desenvolvimento de software.

À medida que os projetos ágeis se tornam maiores e mais complexos, torna-se essencial o uso de ferramentas especializadas para gerenciar e compartilhar informações de forma rápida e eficiente. Atualmente, existe uma variedade de ferramentas comerciais e *opensource* e *freemiums* para o GAP, que vão desde aplicativos sofisticados baseados na web até softwares simples e especializados. Com essas ferramentas é possível reduzir os custos, facilitar a comunicação entre os membros da equipe, possuir um tipo de linguagem acessível, utilizar em conjunto com outras ferramentas na correção de erros, entre outros, de modo a agilizar a entrega do produto final ao cliente.

Conforme apresentado no Capítulo 2, dentre a vasta quantidade de ferramentas que o mercado possui, foi possível perceber que ferramentas com muitos recursos são proprietárias, ou seja, dependem de pagamento para utilização. O que muitas vezes significa que a quantidade de recursos está atrelada às faixas de preços, ou seja, quanto mais recursos, maior o preço. Por outro lado, a minoria das ferramentas é de utilização gratuita, e muitas delas com quantidade limitada de recursos. Além do fator preço, outro ponto está relacionado à complexidade das ferramentas, uma vez que algumas não são intuitivas e não são fáceis de serem utilizadas.

Este trabalho apresenta uma análise de 22 ferramentas *opensources*, *freemiums* e pagas com teste grátics visando auxiliar o processo de seleção de ferramentas para apoiar o GAP. Para avaliar cada uma das ferramentas selecionadas, foram definidas 34 diretrizes a

fim de identificar suas similaridades, diferenças, características, vantagens e desvantagens.

O presente Capítulo apresenta o processo da análise das ferramentas para auxiliar o gerenciamento ágil de projetos de software. Na Seção 3.1 são apresentadas as ferramentas selecionadas e uma visão geral em relação às suas funcionalidades. Na Seção 3.2 é detalhado como as diretrizes foram definidas para auxiliar o processo de análise das ferramentas. Por fim, na Seção 3.3 é detalhado o processo metodológico.

## 3.1 Seleção das ferramentas

A seleção de ferramentas ocorreu a partir dos oito trabalhos relacionados, apresentados na Tabela 3, os quais analisaram um total de 31 ferramentas. A partir dessas ferramentas, foram selecionadas 13 ferramentas que foram abordadas por pelo menos dois estudos relacionados. Para complementar essas ferramentas, foram pesquisados em sites e blogs sobre as ferramentas mais utilizadas no mercado que não foram contempladas pelos estudos. Um dos sites consultados foi o Capterra[1] que continha uma lista das ferramentas, suas especificações, formas de utilização, requisitos e tipo de licença. Neste site, as ferramentas estavam ordenadas considerando a avaliação dos usuários. No entanto, somente era apresentado uma média da avaliação, não indicando o número de usuários que avaliou a ferramenta. A Tabela 4 apresenta a lista de ferramentas analisadas e seus respectivos links de acessos.

A Tabela 5 apresenta as características das 22 ferramentas analisadas agrupadas de acordo com o seu tipo (primeira coluna), nome da ferramenta (segunda coluna), número de estudos relacionados que abordaram a ferramenta (terceira coluna), tipo de licença (quarta coluna) e o tipo de plataforma (quinta coluna). As ferramentas analisadas foram agrupadas em T1, T2 e T3. As ferramentas categorizadas como T1 são aquelas classificadas como *opensource*, T2 como *freemium* e a T3 como *paga com teste grátis* que foram abordadas pelos estudos relacionados (apresentados no Capítulo 2). É importante ressaltar que a análise das ferramentas foi realizada pelo autor do trabalho e que em caso de dúvidas a professora orientadora era contatada.

Portanto, com as ferramentas selecionadas como objeto deste estudo, foi necessário estabelecer critérios e diretrizes que nortearam a análise comparativa entre elas. Para que fosse possível demonstrar as características e as funcionalidades de cada ferramenta.

[1] https://www.capterra.com.br/

**Tabela 4 - Ferramentas analisadas.**

| Nome | Link de acesso |
|---|---|
| Versione | https://www.collab.net/products/versionone |
| Assembla | https://www.assembla.com/home |
| Rally | https://www.broadcom.com |
| TraC | https://trac.edgewall.org |
| Pivotal Tracker | https://www.pivotltracker.com |
| Taiga | https://www.taiga.io |
| ActiveCollab | https://activecollab.com |
| Britix24 | https://www.bitrix24.com.br |
| Microsoft Project | https://www.microsoft.com/pt-br/microsoft-365/project/ |
| Trello | https://trello.com/home |
| Jira | https://www.atlassian.com/software/jira |
| Asana | https://asana.com |
| Azure DevOps | https://azure.microsoft.com/pt-br/services/devops/ |
| Wrike | https://www.wrike.com |
| GanttProject | https://www.ganttproject.biz |
| Artia | https://artia.com |
| Monday | https://monday.com/lang/pt/ |
| Basecamp | https://basecamp.com |
| Slack | https://slack.com/intl/pt-br/ |
| Runrun.it | https://runrun.it/pt-BR |
| Airtable | https://www.airtable.com |
| MeisterTask | https://www.meistertask.com/pt |

## 3.2 Definição dos critérios e diretrizes

Com a seleção das ferramentas, na Tabela 6 são apresentados o ID dos critérios (primeira coluna), a descrição dos critérios definidos e criados (segunda coluna) e, por fim, os estudos dos quais os critérios foram definidos (terceira coluna). Os critérios 5, 9, 10, 32, 33 e 34 foram criados com base nos princípios do GAP apresentados na Tabela 1.

**Tabela 6**: Critérios definidos com base nos trabalhos relacionados.

| ID | Critério | Estudo |
|---|---|---|
| 1 | Necessidade do negócio | Mihalache (2017) |
| 2 | Gerenciamento de projetos | Oliveira (2010) |
| 3 | Coleta dos requisitos | Bajwa e Kaur (2017) |
| 4 | Estórias dos usuários | Mihalache (2017) |
| 5 | Priorização dos requisitos | - |
| 6 | Criação de tarefas | Lima (2021) |

Continuação na próxima página

**Tabela 6 – continuação da página anterior.**

| ID | Diretriz | Estudo |
|---|---|---|
| 7 | Controle de tarefas | Manole e Avramescu (2017) |
| 8 | Gerenciamento do escopo do projeto | Oliveira (2010) |
| 9 | Definição do marco do projeto | - |
| 10 | Estimativa de duração das tarefas | - |
| 11 | Visualização do progresso das tarefas | Mihalache (2017) |
| 12 | Desenvolvimento do cronograma | Özkan e Mishra (2019), Bajwa e Kaur (2017), Mihalache (2017) |
| 13 | Gerenciamento do tempo | Oliveira (2010) |
| 14 | Criação do gráfico de Gantt | Bajwa e Kaur (2017) |
| 15 | Registro dos testes das tarefas | Özkan e Mishra (2019) |
| 16 | Identificação de defeitos | Bajwa e Kaur (2017) , Özkan e Mishra (2019) |
| 17 | Identificação de problemas de não conformidade | Buturugã, Gogoi e Prodan (2016) |
| 18 | Gerenciamento da qualidade | Oliveira (2010) |
| 19 | Realização da comunicação | Buturugã, Gogoi e Prodan (2016) |
| 20 | Troca de mensagens entre os usuários | Lima (2021) |
| 21 | Notificações por e-mail | Bajwa e Kaur (2017) |
| 22 | Gerenciamento de comunicação | Oliveira (2010) |
| 23 | Identificação de possíveis riscos | Mihalache (2017) |
| 24 | Identificação de soluções | Mihalache (2017) |
| 25 | Permitir o gerenciamento de risco | Oliveira (2010) |
| 26 | Geração de relatórios | Özkan e Mishra (2019), (BUTURUGã; GOGOI; PRODAN, 2016), Azizyan (2011) |
| 27 | Geração de documentos | Özkan e Mishra (2019) |
| 28 | Criação do gráfico *burndown* | Özkan e Mishra (2019) |
| 29 | Criação, configuração e manipulação de múltiplos projetos | Bajwa e Kaur (2017) |
| 30 | Controle de versão | Özkan e Mishra (2019) |

Continuação na próxima página

**Tabela 6 – continuação da página anterior.**

| ID | Diretriz | Estudo |
|---|---|---|
| 31 | Integração com outros programas | Özkan e Mishra (2019), Manole e Avramescu (2017), Bajwa e Kaur (2017), Azizyan (2011) |
| 32 | Gerenciamento de documentos | - |
| 33 | Gerenciamento de arquivos | - |
| 34 | Controle de marcos | - |

As diretrizes do presente trabalho foram definidas com base na análise dos critérios utilizados nos estudos relacionados apresentados na Tabela 6. Essas diretrizes foram agrupadas em oito categorias (C = $c_1$, $c_2$, ..., $c_8$) sendo que as seis primeiras contemplam as áreas de conhecimento do PMBOK (PMI, 2017). As categorias são:

- **$C_1$ - Integração:** inclui os processos e as atividades necessárias para identificar, definir, combinar, unificar e coordenar o gerenciamento de projetos;
- **$C_2$ - Escopo:** visa assegurar que o projeto contemple todo o trabalho necessário, e apenas o necessário, para que o mesmo termine com sucesso;
- **$C_3$ - Cronograma:** visa gerenciar o término pontual do projeto;
- **$C_4$ - Qualidade:** consiste na incorporação da política de qualidade da organização com relação ao planejamento, gerenciamento e controle dos requisitos de qualidade do projeto e do produto para atender as expectativas das partes interessadas;
- **$C_5$ - Comunicação:** assegurar que as informações do projeto sejam planejadas, coletadas, criadas, distribuídas, armazenadas, recuperadas, gerenciadas, controladas, monitoradas e finalmente organizadas de maneira oportuna e apropriada;
- **$C_6$ - Risco:** consiste na condução de planejamento, identificação e análise de gerenciamento de risco, planejamento de resposta, implementação de resposta e monitoramento de risco em um projeto;
- **$C_7$ - Geração de artefatos:** consiste na criação de artefatos como gráficos, documentos, relatórios, entre outros, que auxiliam no processo de gerenciamento de projetos;

**Tabela 5 - Características das 22 ferramentas analisadas.**

| Tipo | Nome | NE | Licença | Plataforma |
|---|---|---|---|---|
| T1 | Trac | 2 | *Opensource* | Web |
| | Taiga | 2 | *Opensource* | Web e Mobile |
| T2 | Versione | 5 | *Freemium* | Web |
| | Pivotal Tracker | 2 | *Freemium* | Desktop e Mobile |
| | ActiveCollab | 2 | *Freemium* | Desktop e Mobile |
| | Britix24 | 1 | *Freemium* | Desktop e Mobile |
| | Trello | 4 | *Freemium* | Web e Mobile |
| | Asana | 3 | *Freemium* | Web, Mobile e Desktop |
| | Wrike | 1 | *Freemium* | Web, Mobile e Desktop |
| | GanttProject | 1 | *Freemium* | Desktop |
| | Monday | 0 | *Freemium* | Web, Mobile e Desktop |
| | Artia | 0 | *Freemium* | Web e Desktop |
| | Basecamp | 0 | *Freemium* | Web, Mobile e Desktop |
| | Slack | 0 | *Freemium* | Web, Mobile e Desktop |
| | Runrun.it | 0 | *Freemium* | Web e Mobile |
| | Airtable | 0 | *Freemium* | Web, Mobile e Desktop |
| | MeisterTask | 0 | *Freemium* | Web, Mobile e Desktop |
| T3 | Assembla | 3 | Paga com teste grátis | Desktop e Mobile |
| | Rally | 3 | Paga com teste grátis | Web |
| | Microsoft Project | 1 | Paga com teste grátis | Web e Desktop |
| | Jira | 6 | Paga com teste grátis | Web, Mobile e Desktop |
| | Azure DevOps | 2 | Paga com teste grátis | Web e Desktop |

- **$C_8$ - Gerenciamento:** consiste no gerenciamento de artefatos, documentos, arquivos, diferentes projetos, controle de versão, entre outros.

Portanto, foram analisados todos os critérios com o intuito de verificar quais poderiam ser utilizados e transformados em diretrizes no presente trabalho. Consequentemente, algumas diretrizes foram incluídas e outras removidas. A definição das diretrizes foi realizada em três passos:

1. Todos os critérios abordados pelos autores dos trabalhos relacionados foram analisados, para definir quais poderiam ser utilizados e quais não;
2. Os critérios extraídos foram transformados nas diretrizes que orientam este trabalho para que as diretrizes pudessem ser avaliadas por meio de 'Sim" caso a diretriz tenha sido contemplada e "Não" caso contrário.
3. As diretrizes foram agrupadas em categorias;

As 34 diretrizes utilizadas para a avaliação e a comparação das ferramentas são apresentadas na Tabela 7. Essas diretrizes foram agrupadas em oito categorias (primeira

coluna), na segunda coluna é apresentado o ID da diretriz, na terceira coluna a descrição das diretrizes e, por fim, na quarta coluna é apresentado o estudo no qual a diretriz foi retirada. É importante destacar que as diretrizes $D_5$, $D_9$, $D_{10}$, $D_{32}$, $D_{33}$ e $D_{34}$ foram criadas, uma vez que não foram contempladas em nenhum estudo.

**Tabela 7**: Diretrizes definidas e utilizadas na avaliação das 22 ferramentas.

| Categoria | ID | Diretriz | Estudo |
|---|---|---|---|
| Integração | D1 | Permitir registrar a necessidade do negócio | Mihalache (2017) |
| | D2 | Permitir o gerenciamento de projetos | Oliveira (2010) |
| Escopo | D3 | Realizar a coleta dos requisitos | Bajwa e Kaur (2017) |
| | D4 | Uso de estórias dos usuários para coleta de requisitos | Mihalache (2017) |
| | D5 | Permitir a priorização dos requisitos | - |
| | D6 | Permitir a criação de tarefas | Lima (2021) |
| | D7 | Permitir o controle de tarefas | Manole e Avramescu (2017) |
| | D8 | Permitir o gerenciamento do escopo do projeto | Oliveira (2010) |
| Cronograma | D9 | Permitir a definição do marco do projeto | - |
| | D10 | Realizar a estimativa de duração das tarefas | - |
| | D11 | Permitir visualizar o progresso das tarefas | Mihalache (2017) |
| | D12 | Realizar o desenvolvimento do cronograma | Özkan e Mishra (2019), Bajwa e Kaur (2017), Mihalache (2017) |
| | D13 | Permitir o gerenciamento do tempo | Oliveira (2010) |

Continuação na próxima página

**Tabela 7 – continuação da página anterior.**

| Categoria | ID | Diretriz | Estudo |
|---|---|---|---|
| | D14 | Permitir a criação do gráfico de Gantt | Bajwa e Kaur (2017) |
| Qualidade | D15 | Permitir o registro dos testes das tarefas | Özkan e Mishra (2019) |
| | D16 | Permitir a identificação de defeitos | Bajwa e Kaur (2017) , Özkan e Mishra (2019) |
| | D17 | Permitir a identificação de problemas de não conformidade | Buturugã, Gogoi e Prodan (2016) |
| | D18 | Permitir o gerenciamento da qualidade | Oliveira (2010) |
| Comunicação | D19 | Definir como a comunicação será realizada | Buturugã, Gogoi e Prodan (2016) |
| | D20 | Possibilitar a troca de mensagens entre os usuários | Lima (2021) |
| | D21 | Realizar notificações por e-mail | Bajwa e Kaur (2017) |
| | D22 | Permitir o gerenciamento de comunicação | Oliveira (2010) |
| Risco | D23 | Realizar a identificação de possíveis riscos | Mihalache (2017) |
| | D24 | Realizar a identificação de soluções | Mihalache (2017) |
| | D25 | Permitir o gerenciamento de risco | Oliveira (2010) |
| Geração de artefatos | D26 | Permitir a geração de relatórios | Özkan e Mishra (2019), (BUTURUGã; GOGOI; PRODAN, 2016), Azizyan (2011) |
| | D27 | Permitir a geração de documentos | Özkan e Mishra (2019) |
| | D28 | Permitir a criação do gráfico *burndown* | Özkan e Mishra (2019) |
| Gerenciamento | D29 | Permitir a criação, configuração e manipulação de múltiplos projetos | Bajwa e Kaur (2017) |

Continuação na próxima página

**Tabela 7 – continuação da página anterior.**

| Categoria | ID | Diretriz | Estudo |
|---|---|---|---|
| | D30 | Permitir realizar controle de versão | Özkan e Mishra (2019) |
| | D31 | Permitir realizar a integração com outros programas | Özkan e Mishra (2019), Manole e Avramescu (2017), Bajwa e Kaur (2017), Azizyan (2011) |
| | D32 | Permitir o gerenciamento de documentos | - |
| | D33 | Permitir o gerenciamento de arquivos | - |
| | D34 | Permitir controlar marcos | - |

As diretrizes definidas juntamente com as suas categorias foram associadas aos princípios do GAP, definidos na Tabela 1, a fim de confirmar a importância e veracidade das diretrizes para avaliar as ferramentas. Na Tabela 8 são apresentadas as categorias (primeira coluna), as diretrizes (segunda coluna) e os respectivos princípios associados (terceira coluna).

A maior parte das diretrizes foram definidas baseadas nos critérios dos trabalhos relacionados, mas percebeu-se a necessidade de incluir mais algumas diretrizes que esses critérios não abordavam. Conforme apresentado na Tabela 8, as seis diretrizes criadas ($D_5$, $D_9$, $D_{10}$, $D_{32}$, $D_{33}$, $D_{34}$) foram baseadas nos princípios do GAP para que fosse possível ampliar a visão que se tem das ferramentas a partir da perspectiva ágil para projetos.

## 3.3 Metodologia

Neste trabalho foi realizada uma pesquisa básica cuja a finalidade é gerar conhecimentos novos úteis para o avanço da ciência sem aplicação prática prevista. Quanto aos objetivos, a pesquisa foi classificada como exploratória, pois visou conhecer os fatos e fenômenos relacionados às ferramentas para auxiliar o GAP baseado em dados de natureza qualitativa. Para realizar a análise comparativa foi utilizado como procedimento a pesquisa documental a partir de dados coletados por meio da revisão bibliográfica (WAZLAWICK, 2014).

**Tabela 8 - Agrupamento das categorias e diretrizes de acordo com os princípios do GAP.**

| Categorias | Diretrizes | Princípios GAP |
|---|---|---|
| Integração | D1, D2 | P2 - Flexibilidade, P8 - Inovação e criatividade |
| Escopo | D3, D4, D5, D6, D7, D8 | P1- Simplicidade, P3 - Busca por excelência técnica, P4 - Agregar valor para o cliente e a equipe de projetos, P5 - Iterações e entregas parciais, P7 - Tomada de decisão participativa |
| Cronograma | D9, D10, D11, D12, D13, D14 | P2 - Flexibilidade, P3 - Busca por excelência técnica, P4 - Agregar valor para o cliente e a equipe de projetos, P6 - Autogestão e auto-organização, P7 - Tomada de decisão participativa |
| Qualidade | D15, D16, D17, D18 | P3 - Busca por excelência técnica |
| Comunicação | D19, D20, D21, D22 | P9 - Comunicação |
| Risco | D23, D24, D25 | P3 - Busca por excelência técnica, P7 - Tomada de decisão participativa |
| Geração de Artefatos | D26, D27, D28 | P1 - Simplicidade, P2 - Flexibilidade |
| Gerenciamento | D29, D30, D31, D32, D33, D34 | P3 - Busca por excelência técnica, P4 - Agregar valor para o cliente e a equipe de projetos, P5 - Iterações e entregas parciais, P6 - Autogestão e auto-organização, P7 - Tomada de decisão participativa |

**Fonte: Autoria Própria.**

Para a conclusão dos objetivos e a realização desta pesquisa, foram realizadas atividades previamente estabelecidas. Essas atividades são apresentadas na Figura 5.

- **Atv. 1 - Levantamento dos aspectos conceituais:** nesta atividade foi realizado um estudo sobre os conceitos relacionados ao projeto por meio de uma revisão narrativa, bem como a identificação de trabalhos relacionados, conforme apresentado no Capítulo 2;

- **Atv. 2 - Investigação das ferramentas:** esta atividade consistiu na identificação das ferramentas que auxiliam o gerenciamento ágil de projetos. Essas ferramentas foram identificadas na Seção 2.4 do Capítulo 2, bem como ferramentas comerciais;

- **Atv. 3 - Seleção das ferramentas:** com base na investigação realizada na Atv. 2,

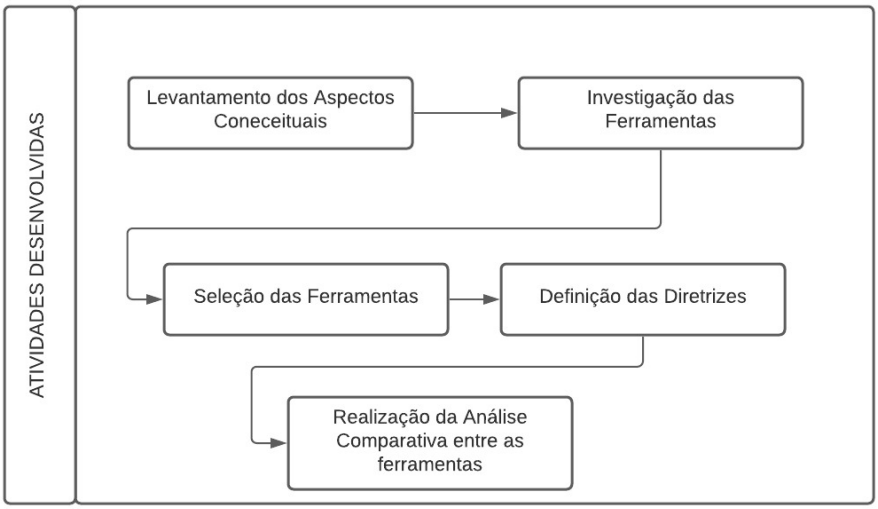


**Figura 5 - Visão geral do processo metodológico.**
**Fonte: Autoria própria.**

foram selecionadas 22 ferramenta para análise;

- **Atv. 5 - Definição das diretrizes:** as 34 diretrizes foram definidas e criadas com base nas principais características e funcionalidades das ferramentas apresentadas na Seção 2.4 do Capítulo 2;
- **Atv. 6 - Realização da análise comparativa entre as ferramentas:** foi realizada uma análise individual de cada ferramenta e, em seguida, a comparação entre elas com base nos diretrizes definidas na Atv. 5.

# 4 RESULTADOS

O presente Capítulo apresenta os resultados alcançados a partir da da análise das 22 ferramentas para auxiliar o gerenciamento ágil de projetos de software. Considerando as 34 diretrizes propostas e descritas nas Tabela 7, foi realizada a análise das 22 ferramentas. A Tabela 9 apresenta quais as diretrizes que são contempladas por cada ferramenta.

**Tabela 9**: Diretrizes contempladas por cada ferramenta.

| **Ferramentas** | **Diretrizes contempladas** | **Total** |
|---|---|---|
| Versione | D2, D5, D6, D7, D9, D12, D14, D15, D17, D27, D28 | 11 |
| Assembla | D5, D6, D7, D9, D11, D12, D14, D15, D18, D26, D30, D32, D34 | 13 |
| Rally | D1, D6, D7, D10, D11, D12, D14, D15, D17, D29, D31 | 11 |
| Trac | D1, D2, D3, D7, D9, D16, D17, D18, D24, D27, D34 | 11 |
| Pivotal Tracker | D4, D6, D7, D9, D10, D11, D12, D15, D17,D26, D30 | 11 |
| Taiga | D2, D4, D5, D6, D7, D9, D11, D13, D17, D18, D26, D27, D31, D32, D33, D34 | 16 |
| ActiveCollab | D2, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15, D17, D18, D19, D20, D21, D22, D26, D27, D30, D31, D32, D33 | 23 |
| Britix24 | D1, D2, D3, D5, D6, D7, D8, D9, D10, D11, D13, D14, D15, D19, D20, D21, D22, D26, D27, D29, D30, D31, D32, D33, D34 | 25 |

Continuação na próxima página

**Tabela 9 – continuação da página anterior.**

| Ferramentas | Diretrizes contempladas | Total |
|---|---|---|
| Microsoft Project | D1, D2, D3, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15, D17, D18, D19, D22, D23, D24, D25, D26, D28, D29, D32, D33, D34 | 26 |
| Trello | D1, D2, D3, D4, D5, D6, D7, D8, D9, D11, D12, D13, D14, D17, D18, D19, D21, D22, D23, D25, D26, D27, D29, D32, D33, D34 | 26 |
| Jira | D1, D2, D3, D5, D6, D7, D8, D9, D11, D12, D13, D14, D17, D18, D19, D22, D26, D27, D31, D32, D34 | 21 |
| Asana | D1, D2, D3, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15, D16, D17, D18, D19, D20, D21, D22, D23, D25, D26, D27, D29, D30, D31, D32, D33, D34 | 31 |
| Azure DevOps | D1, D2, D3, D5, D6, D8, D14, D19, D22, D24, D26, D27, D30, D31, D32, D33 | 16 |
| GanttProject | D2,D5, D8, D9, D13, D14, D17, D34 | 8 |
| Artia | D1, D2, D5, D6, D7, D8, D9, D10, D11, D13, D14, D15, D17, D19, D20, D22, D23, D25, D26, D27, D29, D31, D32, D33, D34 | 25 |
| Monday | D1, D2, D3, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15, D18, D19, D20, D21, D22, D23, D25, D26, D27, D28, D30, D31, D32, D33, D34 | 29 |
| Basecamp | D1, D2, D3, D5, D6, D7, D8, D10, D11, D13, D15, D17, D19, D20, D26, D27, D29, D31, D32, D33 | 20 |
| Slack | D1, D2, D3, D5, D6, D8, D11, D13, D15, D18, D19, D20, D22, D27, D31, D32, D34 | 17 |
| Runrun.it | D1, D2, D3, D5, D6, D7, D8, D10, D11, D13, D14, D15, D19, D22, D24, D26, D27, D31, D32 | 19 |
| Wrike | D1, D2, D3, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15, D17, D18, D19, D21, D22, D23,D25, D26, D27, D29, D30, D31, D32, D33, D34 | 29 |

Continuação na próxima página

**Tabela 9 – continuação da página anterior.**

| Ferramentas | Diretrizes contempladas | Total |
|---|---|---|
| Airtable | D1, D2, D3, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15, D17, D18, D19,D20, D21, D22, D23, D26, D27, D28, D29, D30, D31, D32, D33, D34 | 30 |
| MeisterTask | D1, D2, D3, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15, D17, D18, D19, D21, D22, D23, D26, D27, D28, D29, D31, D32, D33, D34 | 28 |

De acordo com os resultados, é possível notar que dentre as ferramentas selecionadas a partir dos estudos (T1) a *Asana* contemplou 91% (31/34) das diretrizes, enquanto que entre as ferramentas não citadas pelos estudos (T2) como a *Airtable* contemplou 88% (30/34) das diretrizes e as ferramentas *Wrike* e *Monday* contemplaram 85% (29/34) das diretrizes.

Por outro lado, as ferramentas *VersionOne*, *Pivotal Tracker*, *Rally* e *Trac* contemplaram 32% (11/34) das diretrizes, enquanto que ferramenta *GanttProject* abordou somente 23% (8/34). Entre as ferramentas agrupadas em T2, a ferramenta *Slack* contemplou 50% (17/34) das diretrizes.

A Figura 6 apresenta as diretrizes que foram e não foram contempladas pelas 22 ferramentas. Para indicar que uma determinada diretriz foi contemplada por uma ferramenta é utilizada a letra "S" que representa "Sim", caso contrário, diretriz não contemplada, é utilizada a letra "N" que representa "Não".

De acordo com os resultados, somente quatro diretrizes foram contempladas por mais de 80% das ferramentas. A diretriz "$D_6$ - Permitir a criação de tarefas" pertencente à categoria "$C_2$ - Escopo" foi contemplada por 90% (20/22) das ferramentas; as diretrizes "$D_2$ - Permitir o gerenciamento de projetos pertencente a categoria "$C_1$ - Integração" e "$D_7$ - Permitir o gerenciamento (controle) de tarefas" pertencente a categoria "$C_2$ - Escopo" foram contempladas por 86% (19/22); e por fim, a diretriz "$D_{11}$ - Permitir visualizar o progresso das tarefas" pertencente a categoria "$C_3$ - Cronograma" foi contemplada por 82% (18/22).

Por outro lado, três diretrizes foram contempladas por menos de 20% das

| FERRAMENTAS | C1 | | C2 | | | | | | C3 | | | | | | C4 | | | | C5 | | | | C6 | | | C7 | | | C8 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 | D10 | D11 | D12 | D13 | D14 | D15 | D16 | D17 | D18 | D19 | D20 | D21 | D22 | D23 | D24 | D25 | D26 | D27 | D28 | D29 | D30 | D31 | D32 | D33 | D34 |
| VersionOne | N | S | N | N | S | S | S | N | S | N | N | S | N | S | S | N | S | N | N | N | N | N | N | N | N | N | S | S | N | N | N | N | N | N |
| Assembla | N | N | N | N | S | S | S | N | S | N | S | S | N | S | S | N | N | S | N | N | N | N | N | N | N | S | N | N | N | S | N | S | N | S |
| Rally | S | N | N | N | N | S | S | N | N | S | S | S | N | S | S | N | S | N | N | N | N | N | N | N | N | N | N | N | S | N | S | N | N | N |
| Trac | S | S | S | N | N | N | S | N | S | N | N | N | N | N | N | S | S | S | N | N | N | N | N | S | N | N | S | N | N | N | N | N | N | S |
| Pivotal Tracker | N | N | N | S | N | S | S | N | S | S | S | S | N | N | S | N | S | N | N | N | N | N | N | N | N | S | N | N | N | S | N | N | N | N |
| Taiga | N | S | N | S | S | S | S | N | S | N | S | N | S | N | N | N | S | S | N | N | N | N | N | N | N | S | S | N | N | N | S | S | S | S |
| ActiveCollab | N | S | N | N | N | S | S | S | S | S | S | S | S | S | S | N | S | S | S | S | S | S | N | N | N | S | S | N | N | S | S | S | S | N |
| Britix24 | S | S | S | N | S | S | S | S | S | S | S | N | S | S | S | N | N | N | S | S | S | S | N | N | N | S | S | N | S | S | S | S | S | S |
| Microsoft Project | S | S | S | N | N | S | S | S | S | S | S | S | S | S | S | N | S | S | S | N | N | S | S | S | S | S | N | S | S | N | N | S | S | S |
| Trello | S | S | S | S | S | S | S | S | S | N | S | S | S | S | N | N | S | S | S | N | S | S | S | N | S | S | S | N | S | N | N | S | S | S |
| Jira | S | S | S | N | S | S | S | S | S | N | S | S | S | S | N | N | S | S | S | N | N | S | N | N | N | S | S | N | N | N | S | S | N | S |
| Asana | S | S | S | N | S | S | S | S | S | S | S | S | S | S | S | S | S | S | S | S | S | S | S | N | S | S | S | N | S | S | S | S | S | S |
| Azure DevOps | S | S | S | N | S | S | N | S | N | N | N | N | N | S | N | N | N | N | S | N | N | S | N | S | N | S | S | N | N | S | S | S | S | N |
| GanttProject | N | S | N | N | S | N | N | S | S | N | N | N | S | S | N | N | S | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | S |
| Artia | S | S | N | N | S | S | S | S | S | S | S | N | S | S | S | N | S | N | S | S | N | S | S | N | S | S | S | N | S | N | S | S | S | S |
| Monday | S | S | S | N | S | S | S | S | S | S | S | S | S | S | S | N | N | S | S | S | S | S | S | N | S | S | S | S | N | S | S | S | S | S |
| Basecamp | S | S | S | N | S | S | S | S | N | S | S | N | S | N | S | N | S | N | S | S | N | N | N | N | N | S | S | N | S | N | S | S | S | N |
| Slack | S | S | S | N | S | S | N | S | N | N | S | N | S | N | S | N | N | S | S | S | N | S | N | N | N | N | S | N | N | N | S | S | N | S |
| Runrun.it | S | S | S | N | S | S | S | S | N | S | S | N | S | S | S | N | N | N | S | N | N | S | N | S | N | S | S | N | N | N | S | S | N | N |
| Wrike | S | S | S | N | S | S | S | S | S | S | S | S | S | S | S | N | S | S | S | N | S | S | S | N | S | S | S | N | S | S | S | S | S | S |
| Air Table | S | S | S | N | S | S | S | S | S | S | S | S | S | S | S | N | S | S | S | S | S | S | S | N | N | S | S | S | S | S | S | S | S | S |
| MeisterTask | S | S | S | N | S | S | S | S | S | S | S | S | S | S | S | N | S | S | S | N | S | S | S | N | N | S | S | S | S | N | S | S | S | S |
| TOTAL | 16 | 19 | 14 | 3 | 17 | 20 | 19 | 16 | 17 | 13 | 18 | 13 | 16 | 17 | 16 | 2 | 16 | 13 | 15 | 8 | 8 | 14 | 8 | 4 | 6 | 17 | 17 | 5 | 10 | 9 | 15 | 17 | 13 | 15 |

**Figura 6 - Diretrizes contempladas e não contempladas por cada ferramenta. Fonte: Autoria própria.**

ferramentas. A diretriz "$D_{16}$ - Permitir a identificação de bugs" pertencente à categoria "$C_4$ - Qualidade" foi contemplada somente por 9% (2/22) das ferramentas (*Asana e Trac*); a diretriz "$D_4$ - Uso de Estórias dos usuários para coleta de requisitos" pertencente à categoria "$C_2$ - Escopo" foi contemplada somente por 14% (3/22) das ferramentas (*Taiga, Trello, Pivotal Tracker*); e por fim, a diretriz "$D_{24}$ - Realizar a identificação de soluções" pertencente à categoria "$C_6$ - Risco" foi contemplada somente por 18% (4/22) das ferramentas (*Runrun.it, Azure, Trac, Microsoft Project*).

É importante ressaltar que as diretrizes criadas ("$D_5$ - Permitir a priorização dos requisitos, "$D_9$ - Permitir a definição do marco do projeto", "$D_{10}$ - Realizar a estimativa de duração das tarefas", "$D_{32}$ - Permitir o gerenciamento de documentos", "$D_{33}$ - Permitir o gerenciamento de arquivos" e "$D_{34}$ - Permitir controlar marcos") foram contempladas por 59% das ferramentas, indicando a importância dessas diretrizes no gerenciamento ágil de projeto. É possível notar na Figura 6 que algumas diretrizes não foram contempladas por algumas ferramentas. Essas diretrizes são listadas a seguir:

- As diretrizes "$D_1$ - Permitir registrar a necessidade do negócio" a "$D_2$ - Permitir o gerenciamento de projetos", pertencentes a categoria "$C_1$ - Integração", não foram contempladas por 18% (2/22) das ferramentas (*Assembla* e *Pivotal Tracker*).

- As diretrizes "$D_{15}$ - Permitir o registro dos testes das tarefas", "$D_{16}$ - Permitir a identificação de defeitos", "$D_{17}$ - Permitir a identificação de problemas de não conformidade" e "$D_{18}$ - Permitir o gerenciamento da qualidade", pertencentes a categoria "$C_4$ - Qualidade", não foram contempladas por 4% (1/22) das ferramentas (*AzureDevOps*).

- As diretrizes "$D_{19}$ - Definir como a comunicação será realizada", "$D_{20}$ - Possibilitar a troca de mensagens entre os usuários", "$D_{21}$ - Realizar notificações por e-mail" e "$D_{22}$ - Permitir o gerenciamento de comunicação", pertencentes a categoria, "$C_5$ - Comunicação" não foram contempladas por 32% (7/22) das ferramentas (*Versione, Assembla, Rally, Trac, Pivotal Tracker, Taiga, GanttProject*).

- As diretrizes "$D_{23}$ - Realizar a identificação de possíveis riscos", "$D_{24}$ - Realizar a identificação de soluções" e "$D_{25}$ - Permitir o gerenciamento de risco", pertencentes a categoria "$C_6$ - Risco", não foram contempladas por 54% (12/22) das ferramentas (*Versione, Assembla, Rally, Trac, Pivotal Tracker, Taiga, ActiveCollab, Britix, Jira, GanttProject, Slack e Basecamp*).

- As diretriz "$D_{26}$ - Permitir a geração de relatórios", "$D_{27}$ - Permitir a geração de documentos"e"$D_{28}$ - Permitir a criação do gráfico burndown", pertencentes à categoria "$C_7$ - Geração de artefatos", não foram contempladas por 18% (2/22) das ferramentas (*Rally e GanttProject*).

Por fim, as ferramentas foram analisadas considerando as categorias. Na Figura 7 é possível visualizar o total de diretrizes contempladas por cada ferramenta de acordo com as categorias.

Analisando as categorias, é possível notar que para a categoria de "$C_1$ - Integração" das duas diretrizes relacionadas, em média, as duas foram contempladas pelas ferramentas. Já as categorias "$C_2$ - Escopo", "$C_3$ - Cronograma" e "$C_8$ - Gerenciamento", das seis diretrizes relacionadas em cada categoria, em média, somente quatro foram contempladas pelas ferramentas. É importante destacar que dentre as categorias, as diretrizes pertencentes à categoria "$C_6$ - Risco" foram menos contempladas pelas ferramentas, sendo a diretriz $D_{23}$ contemplada por 36% (8/22) das ferramentas, $D_{24}$ contemplada por 18% (4/22) e $D_{25}$ contemplada por 27% (6/22).

Portanto, as ferramentas, classificadas como T2, *Asana* e *Airtable* conseguiram atender ao maior número das 34 diretrizes utilizadas nesse estudo. Além disso, a diretriz $D_6$ relacionada à categoria "$C_2$ - Escopo", foi contemplada por 20 ferramentas.

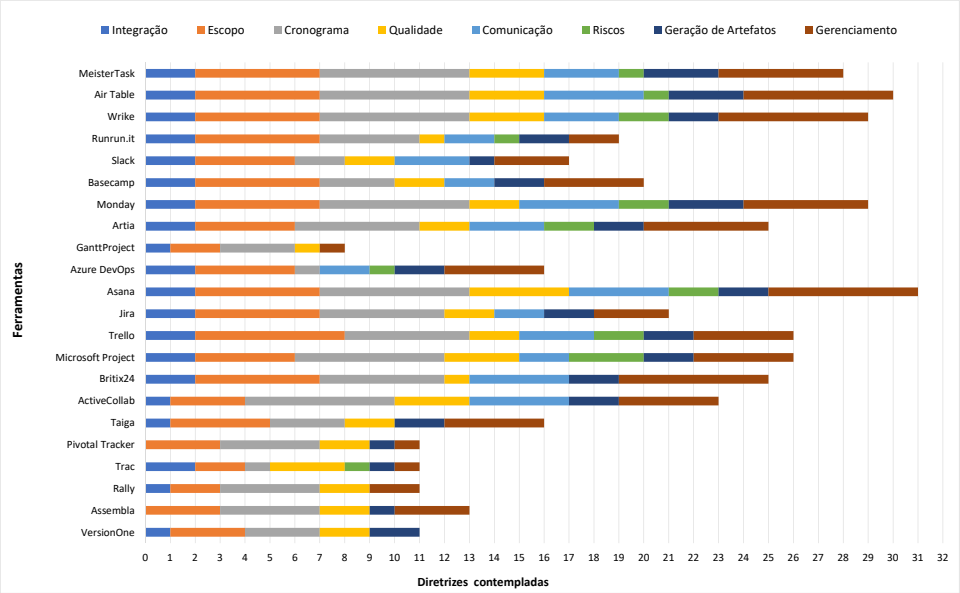


**Figura 7 - Distribuição das diretrizes contempladas pelas ferramentas agrupadas por categoria.**

**Fonte: Autoria própria.**

## 4.1 Ameaças à Validade

As ameaças à validade são influências que podem afetar diretamente as análises conduzidas ao longo do trabalho. É notório que a seleção, busca e análise foi uma parte crucial da avaliação, para a seleção de categorias, diretrizes e estudos já citados, que foram selecionados com base na utilização no mercado de trabalho e de forma acadêmica.

Todavia, é sabido que alguns fatores podem afetar o resultado do trabalho em si, pois foi realizada a análise documental de cada ferramenta, por falta de recursos necessários, como: aplicação, tempo em virtude da pandemia do Covid-19, e a aplicação em empresas que teriam liberação, porém como medida preventiva ao Coronavírus e pelo distanciamento social foi restringido o acesso as empresas.

Portanto, foi possível analisar somente a documentação das ferramentas, para tal, foi levado em consideração o *feedback* de clientes e usuários, para ter uma melhor ideia e de certa forma, poder contemplar e compreender melhor as funcionalidades. No entanto, é sempre importante salientar que, isso pode comprometer a análise, mesmo que seja em uma porcentagem mínima, o risco existe. Com o objetivo de minimizar as ameaças à validade da pesquisa, cada ferramenta foi analisada pelo autor e em casos de dúvidas a professora orientadora foi consultada.

Outro ponto importante, é que para determinar quais diretrizes as ferramentas atingiam, além da documentação, foi considerada as informações do *Capterra*, que é um dos mais importantes disponibilizadores de avaliações de software do mundo, e que apresentam as informações sobre as ferramentas em conjunto com os fornecedores de software, que por sua vez, pagam a plataforma pela oportunidade de venda que ela gera e a mantêm gratuita para consulta do público. As informações não disponíveis na documentação, e nem no *Capterra*, foram coletadas pela orientadora do trabalho.

Por fim, pôde-se perceber que várias ferramentas precisam de constante evolução, devido a demanda do mercado, como é visto tanto na abordagem das categorias apresentadas e nas diretrizes contempladas. Também foi percebido que tais ferramentas, são carentes de suporte e ajuda para utilização, o que tem ocasionado muitas dificuldades para usuários/clientes.

# 5 CONSIDERAÇÕES FINAIS

Partindo do levantamento realizado neste estudo, foi possível identificar diversas ferramentas que auxiliam o GAP. No entanto, apesar da variedade de ferramentas analisadas, é notório que a maioria delas são apenas comerciais. Neste contexto, o objetivo deste estudo foi realizar uma análise comparativa de 22 ferramentas *opensources*, *freemiums* e pagas com teste grátis visando auxiliar o processo de seleção de ferramentas para apoiar o GAP. Para avaliar cada uma das ferramentas selecionadas foram definidas 34 diretrizes a fim de identificar suas similaridades, diferenças e características.

Ao final do desenvolvimento deste estudo, foi possível perceber, através da comparação das ferramentas, um conjunto de evidências que podem vir a auxiliar diferentes usuários na seleção da ferramenta mais adequada, de acordo com as suas necessidades, para apoiar o GAP. Tendo em vista que, algumas ferramentas como por exemplo, *Microsoft Project*, que mesmo sendo paga com teste grátis, para uma imersão de conteúdos e melhor utilização, é necessário pelo menos um mês de utilização da mesma. O mesmo ocorre com outras ferramentas, que mesmo com versões livres, foi notado que é importante o teste de algumas semanas para se ter acesso a grande parte das funcionalidades.

Neste contexto, é importante ressaltar que o trabalho proposto possui algumas limitações assim como ameaças à validade. Essas limitações ocorreram devido a pandemia do Covid-19 que atingiu proporções globais, e problemas que ocasionalmente afetaram a interação com empresas. Em virtude disto, foi decidido pela análise documental das ferramentas bem como avaliações dos usuários, mesmo que sem a instalação da mesma. Esse é um dos pontos abordados e que poderiam indicar uma ameaça à validade, apresentando algumas lacunas referentes a integridade de cada ferramenta.

Ao longo da condução deste trabalho, foi possível observar a grande utilização de ferramentas *opensources*, por serem mais acessíveis economicamente, e que em sua maioria continham outras funções, caso fossem adquiridas suas versões pagas. Outro ponto, é que, de acordo com as diretrizes, várias ferramentas como: *Trac*, *Rally*, *Gantt Project*,

entre outras, contemplam menos diretrizes, o que tende a impactar diretamente com a necessidade do usuário final.

Em contrapartida, as ferramentas *Asana* e *Airtable*, se destacaram, por contemplarem a maior parte das diretrizes, o que permite dizer que atendem muitos requisitos essenciais no gerenciamento de um projeto. Além disso, foi possível observar, outras ferramentas, que contemplam mais de 20 diretrizes, mostrando que tem grande potencial para se tornarem muito promissoras no futuro, como por exemplo, as ferramentas *Trello* e *Microsoft Project*.

Com base nos resultados alcançados, as principais contribuições deste trabalho são:

- Definição de um conjunto de diretrizes para apoiar a avaliação das ferramentas;
- Análise comparativa entre as ferramentas;
- Conjunto de evidências para auxiliar na seleção da ferramenta de Gerenciamento Ágil de Projetos.

# REFERÊNCIAS


AMARAL, D. C. e. a. **Gerenciamento ágil de projetos: aplicação em produtos inovadores**. 6. ed. São Paulo: [s.n.], 2017.

AUGUSTNE, S. **Managing Agile Projects**. 1. ed. Virginia: Prentice Hall PTR, 2005.

AZIZYAN, G. e. a. Survey of agile tool usage and needs. In: **Proceedings of Agile Conference (AGILE)**. [S.l.: s.n.], 2011. p. 29–38.

BAJWA, J. K.; KAUR, J. Comparative study of apm tools. **International Journal of Engineering Science**, v. 24, p. 26–35, 2017.

BECK, K. **eXtreme Programming Explained: Embrace Change.** [S.l.]: Addison Wesley, 1999.

BOEHM, B. Get ready for agile methods, with care. **IEEE Computer Magazine**, v. 35, n. 1, p. 64–69, 2002.

BUTURUGã, O.; GOGOI, V.; PRODAN, I. Agile project management tools. **Informatica Economica**, v. 16, n. 1, p. 19–25, 2016.

CARVALHO, M.; JR., R. R. **Fundamentos em Gestão de Projetos. Construindo Competências Para Gerenciar Projetos**. 4. ed. [S.l.]: Atlas, 2015.

CHIN, G. **Agile Project Management: how to succeed in the face of changing project requirements**. New York: Amacom, 2004.

COCKBURN, B. **Crystal Clear: A Human-Powered Methodology for Small Teams**. [S.l.]: Addison-Wesley Professional, 2004.

COHN, M.; FORD, D. Metodologia fdd aplicada a especificação dos requisitos no processo gre do nível g de maturidade do modelo mps.br. **Introducing an agile process to an organization**, v. 36, p. 74–78, 2003.

CONSORTIUM, A. B. **The DSDM agile project framework Handbook**. 2014. 1–177 p. Disponível em: <`https://fliphtml5.com/glqi/jhlm/basic`>. Acesso em: 10 out. 2020.

GUIDE, A. **Agile Practice Guide / Project Management Institute-PMI e Agile Alliance**. EUA, Pennsylvania: [s.n.], 2017. Disponível em: <`https://www.agilealliance.org/wp-content/uploads/2017/09/AgilePracticeGuide.pdf`>. Acesso em: 10 set. 2020.

HIGHSMITH, J. **Agile Project Management: creating innovative products**. Boston: Addison-Wesley, 2004.

INTERNATIONAL, I. T. S. G. **CHAOS REPORT 2018**. 2018. Disponível em: <https://www.standishgroup.com/sample_research_files/DemoPRBR.pdf>. Acesso em: 08 ago. 2020.

KERZNER, H. **Gestão de Projetos**. [S.l.]: Bookman Editora, 2015.

LEAC, L. **Lean project management: eight principles for success**. Idaho: Advanced Projects Boise, 2005.

LEAL, L. de Q. **Uma abordagem ágil ao gerenciamento de projetos de software baseada no PMBOK Guide**. Tese (Dissertação de Mestrado) — Universidade Federal de Pernambuco (UFPE), Recife, PE, Brasil, Janeiro 2008.

LIMA, K. G. M. **Aplicação de ferramentas gratuitas para o gerenciamento de projetos da equipe titãs da robótica do Ifes Campus Colatina**. Tese (Monografia de Conclusão de Curso) — Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo, Colatina, ES, Brasil, 2021.

MAGALHÃES, A.; ROUILLER, A.; VASCONCELOS, A. O gerenciamento de projetos de software desenvolvidos a luz de metodologias Ágeis: Uma visão comparativa. **Revista Proqualiti**, v. 1, 2005.

MANIFESTO, A. **Manifesto for Agile Software Development**. 2001. Disponível em: <http://www.agilemanifesto.org>. Acesso em: 20 out. 2020.

MANOLE, M.; AVRAMESCU, M. A comparative analysis of agile project management tools. **Informatica Economica**, v. 17, n. 1, p. 25–31, 2017.

MIHALACHE, A. Project management tools for agile teams. **Informatica Economica**, v. 21, n. 4, p. 85–93, 2017.

MISHRA, A.; MISHRA, D. Software project management tools: a brief comparative view. **ACM SIGSOFT Software Engineering Notes**, v. 38, n. 3, p. 1–4, 2013.

OLIVEIRA, M. V. **Análise de ferramentas case de apoio a gestão de TI para atendimento as 9 áreas de gerência de projeto**. Tese (Monografia de Conclusão de Curso) — Faculdades Integradas de Caratinga, Caratinga, MG, Brasil, 2010.

ORACLE. **Custom Development Method Fast Track (CDM Fast Track) Method Handbook, Release 1.2.0**. Parkway, California, USA: Oracle Corporation, 2000.

PMI. **Um guia do conhecimento em gerenciamento de projetos**. 6. ed. [S.l.]: EUA: Project Management Institute, 2017.

PRESSMAN, R. S. **Software engineering: A practitioner's approach**. $8^{th}$. ed. [S.l.]: McGraw-Hill, 2016. 976 p.

PRIKLADNICKI, R.; WILLI, R.; MILANI, F. **Métodos Ágeis para desenvolvimento de software**. Porto Alegre: Bookman, 2014.

SCHWABER, K.; SUTHERLAND, J. **O guia definitivo para o Scrum: As regras do Jogo**. 2017. Disponível em: <https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-Portuguese-European.pdf>. Acesso em: 20 out. 2020.

SILVA, S.; JR, N. P. Metodologias Ágeis para o gerenciamento do desenvolvimento de ead em uma universidade corporativa. **Em Rede - Revista de Educação a Distância**, v. 5, n. 3, p. 584–598, 2018.

SOFTHOUSE. **Scrum in five minutes**. 2014. Disponível em: <https://pt.scribd.com/document/56205069/Softhouse-Scrum-in-5-Minutes>. Acesso em: 12 set. 2020.

SOMMERVILLE, I. **Software Engineering**. $10^{th}$. ed. [S.l.]: Pearson Addison-Wesley, 2016. 816 p.

UDO, N.; KOPPENSTEINER, S. **WILL AGILE DEVELOPMENT CHANGE THE WAY WE MANAGE SOFTWARE PROJECTS? AGILE FROM A PMBOK GUIDE PERSPECTIVE**. 2003. 1–7 p. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.125.3436&rep=rep1&type=pdf. Acesso em: 10 out. 2020.

UDO, N.; KOPPENSTEINER, S. **Will agile development change the way we manage software projects? Agile from a Pmbok guide perspective**. 2003. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.125.3436&rep=rep1&type=pdf>. Acesso em: 10 out. 2021.

WAZLAWICK, R. S. **Metodologia de pesquisa para ciência da computação**. [S.l.]: Elsevier, 2014.

ÖZKAN, D.; MISHRA, A. Agile project management tools: A brief comparative view. **Cybernetics and Information Technologies**, v. 19, n. 4, p. 183–200, 2019.