UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE TECNOLOGIA

Anderson Alves de Carvalho Aguiar

Métricas, metodologias e práticas adaptadas ao gerenciamento da evolução de um portal de auxílio informativo e de desenvolvimento tecnológico

Fortaleza

2023

Anderson Alves de Carvalho Aguiar

Métricas, metodologias e práticas adaptadas ao gerenciamento da evolução de um portal de auxílio informativo e de desenvolvimento tecnológico

Dissertação apresentada ao Programa de Graduação em Engenharia de Computação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de engenheiro em computação.

Orientador: Prof. Dr. José Marques Soares.

Fortaleza

2023

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

A227m Aguiar, Anderson Alves de Carvalho.
Métricas, metodologias e práticas adaptadas ao gerenciamento da evolução de um portal de auxílio informativo e de desenvolvimento tecnológico / Anderson Alves de Carvalho Aguiar. – 2023.
52 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Centro de Tecnologia, Curso de Engenharia de Computação, Fortaleza, 2023.
Orientação: Prof. Dr. José Marques Soares.

1. Métricas. 2. Scrum. 3. Metodologias ágeis. 4. Estudo de caso. I. Título.

CDD 621.39

ANDERSON ALVES DE CARVALHO AGUIAR

Métricas, metodologias e práticas adaptadas ao gerenciamento da evolução de um portal de auxílio informativo e de desenvolvimento tecnológico

Trabalho de conclusão apresentada ao Programa de Graduação em Engenharia de computação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de engenheiro em computação.

Aprovada em: 13 / 07 / 2023.

BANCA EXAMINADORA

_____
Prof. Dr. José Marques Soares (Orientador)
Universidade Federal do Ceará (UFC)

_____
Prof. MSc. Patrícia Jamile de Oliveira Martins
Instituto Federal de Educação, Ciência e Tecnologia do Ceará. IFCE

_____
Prof. Jefferson Calixto Figueiredo
Instituto Federal de Educação, Ciência e Tecnologia do Ceará. IFCE

AGRADECIMENTOS

À Instituição Universidade Federal do Ceará, pelo apoio e oportunidade de adquirir conhecimento e capacitação para me tornar um excelente profissional no mercado de trabalho.

À minha família, em especial a minha mãe, Socorro, uma mulher batalhadora e um exemplo de mulher e de força que formou-se em sua segunda graduação aos 62 anos de idade e sempre nos ensinou que o mais valioso que podemos deixar para nosso filho são os estudos e o conhecimento. Hoje, não esta entre nós mas sigo batalhando e lembrando dos seus sábios ensinamentos, sei que parte da força de viver e de aprender herdei com muito orgulho dela.

À vida e a Deus, que durante os últimos anos me proporcionou uma segunda chance de viver. Hoje, lembro que neste mesmo período estava concluindo um tratamento de quimioterapia para combater um linfoma, apesar do processo doloroso consegui resignificar esta experiência e com isso obter muitos ensinamentos e me transformar em um amante pela vida e seus mistérios, serei eternamente grato por esta experiência. Gostaria de agradecer à todos que participaram e me auxiliaram neste tratamento, são tantas pessoal que ficará difícil citar sem passar páginas e páginas agradecendo.

Ao Prof. Dr. José Marques Soares, pela excelente orientação e pela orientação de meu estágio também. Tenho o senhor como referência de profissional e pessoa a ser seguido. Muito obrigado também pelo apoio durante meu tratamento.

Aos professores participantes da banca examinadora pelo tempo, pelas valiosas colaborações e sugestões.

Meus agradecimentos será um sucinto pois não sou muito bom com palavras.

“Ciência e tecnologia revolucionam nossas vidas, mas a memória, a tradição e o mito moldam nossas respostas”

Arthur Schlesinger

RESUMO

Este trabalho tem como objetivo explorar de forma abrangente as métricas, metodologias e práticas na gestão de projetos de TI, com foco especial nas metodologias ágeis, como o Scrum. São examinados os desafios enfrentados na gestão de projetos de desenvolvimento de software e as melhores práticas para superá-los, levando em consideração a utilização de métricas para avaliar o desempenho e o progresso dos projetos.

Ao longo deste trabalho acadêmico, são explorados um estudo de caso, exemplos práticos e referências bibliográficas que demonstram a importância das métricas, metodologias e práticas na gestão de projetos de TI. São discutidos os benefícios da adoção de metodologias ágeis, como o Scrum, e como as métricas podem ser utilizadas de forma eficaz para garantir o sucesso dos projetos de desenvolvimento de software. Possuindo diversos desafios, entre eles a comparação da aplicação dos modelos teóricos e práticos e seus impactos em ambientes reais.

Este trabalho tem como resultados a construção de métricas de acordo com o contexto real e com o intuito de aprimorar as práticas e tomadas de decisão durante a gestão de um projeto de desenvolvimento de software.

Palavras-chave: Métricas; Scrum; Metodologias ágeis.

ABSTRACT

This work aims to comprehensively explore metrics, methodologies, and practices in IT project management, with a special focus on agile methodologies, such as Scrum. The challenges faced in the management of software development projects and the best practices to overcome them will be examined, considering the use of metrics to evaluate the performance and progress of the projects.

Throughout this academic work, a case study, practical examples, and bibliographical references that demonstrate the importance of metrics, methodologies, and practices in the management of IT projects will be explored. The benefits of adopting agile methodologies, such as Scrum, and how metrics can be used effectively to ensure the success of software development projects will be discussed. Possessing several challenges, including the comparison of the application of theoretical and practical models and their impacts in real environments.

This work results in the construction of metrics according to the real context and with the aim of improving practices and decision making during the management of a software development project.

Keywords: Metrics; Scrum; Agile methodologies.

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

TI	Tecnologia da Informação
SP	Story Point
PO	Product Owner
SP_Planned	Story Points Planned
SP_Closed	Story Points Closed
SP_Open	Story Points Open
SP_hisorias_Sprint	Story Points das histórias da Sprint

SUMÁRIO

1 INTRODUÇÃO

A gestão de projetos possui uma história rica e evolutiva que remonta a muitos séculos, com evidências de práticas de planejamento e organização desde as construções complexas das antigas civilizações. No entanto, o campo da gestão de projetos como disciplina formal começou a se desenvolver apenas no século XX. Segundo Kerzner (2017), "a gestão de projetos evoluiu como uma disciplina independente nas últimas décadas, buscando melhorar a eficiência, o controle e o sucesso dos projetos".

Os projetos atualmente estão se tornando cada vez mais complexos, envolvendo diversas partes interessadas, requisitos em constante mudança e tecnologias avançadas. Conforme destacado por Turner e Müller (2014), "a complexidade é um dos principais desafios enfrentados pelos gestores de projetos, exigindo abordagens adaptativas e flexíveis".

No início do século XX, a gestão de projetos começou a ser aplicada em setores específicos, como a construção civil, indústria bélica e de tecnologia. Atualmente a tecnologia desempenha um papel fundamental nas empresas, considerada um dos principais impulsionadores da transformação digital e da competitividade.

Neste contexto, a gestão eficaz de projetos de Tecnologia da Informação (TI) é fundamental para o sucesso e a entrega de valor nos ambientes empresariais modernos. A crescente complexidade e a natureza dinâmica dos projetos de desenvolvimento de software exigem abordagens e metodologias ágeis que possam se adaptar às mudanças constantes e fornecer resultados de alta qualidade de forma eficiente. O Scrum (Scrum Guide 2020), uma das metodologias ágeis mais populares, tem se destacado como uma abordagem eficaz para a gestão de projetos de TI.

O Scrum é um *framework* ágil que oferece uma estrutura flexível e iterativa para o gerenciamento de projetos de tecnologia. Ele se baseia em princípios de transparência, inspeção e adaptação, permitindo que as equipes se auto-organizem e entreguem valor de forma incremental.

Além do Scrum, existem diversas outras metodologias ágeis amplamente utilizadas na gestão de projetos de TI, como o Kanban (RADIGAN, D, 2022), XP (Extreme Programming)(BECK, K.; ANDRES, C, 2004) e Lean (POPPENDIECK, M.; POPPENDIECK, T, 2009). Essas metodologias compartilham princípios fundamentais, como a entrega incremental, o foco no valor do cliente e a busca pela melhoria contínua. A escolha da metodologia adequada dependerá das características e necessidades específicas de cada projeto.

Diante dessas metodologias ágeis, um aspecto crucial na gestão de projetos de TI é a utilização de métricas para monitorar e medir o desempenho, a produtividade e a qualidade do projeto. As métricas fornecem informações valiosas que permitem avaliar o progresso, identificar possíveis problemas e tomar decisões embasadas.

Este trabalho acadêmico tem como importância a exploração das métricas, das metodologias ágeis e das práticas na gestão de projetos de TI. São abordados conceitos relacionados ao Scrum e às métricas em um projetos de desenvolvimento de software. Além disso, é apresentado um estudo de caso para ilustrar a aplicação prática e a adaptação desses conceitos ao processo de evolução de um Portal.

1.1 Objetivos

O objetivo geral deste trabalho é analisar e discutir as métricas, metodologias e práticas na gestão de projetos de TI, com foco no contexto de um projeto de desenvolvimento de software. Pretende-se explorar as análises práticas e os benefícios das metodologias ágeis, como o Scrum, e o uso de métricas para o gerenciamento eficaz desses projetos. Além disso, busca-se aplicar um estudo de caso na compreensão e aplicação dos conceitos teóricos, fornecendo *insights* práticos e exemplos reais.

Como objetivos específicos, elencam-se:

- Analisar as características e princípios do Scrum como uma metodologia ágil para a gestão de projetos de desenvolvimento de software.
- Realizar estudo de caso de projetos de desenvolvimento de software que aplicaram o Scrum e métricas, analisando os resultados, lições aprendidas e melhores práticas.
- Analisar os impactos e implicações práticas das métricas, metodologias e práticas na gestão de projetos de TI, considerando aspectos como qualidade, produtividade, satisfação do cliente e cumprimento dos prazos.
- Propor recomendações e diretrizes para o uso efetivo de métricas e metodologias ágeis na gestão de projetos de desenvolvimento de software, visando melhorar o desempenho e o sucesso dos projetos.

Ao alcançar esses objetivos, espera-se contribuir para a compreensão aprofundada das métricas, metodologias e práticas na gestão de projetos de TI, fornecendo um embasamento

teórico consistente e exemplos práticos de sua aplicação. Isso permitirá que gestores, profissionais e pesquisadores no campo da TI tenham um melhor entendimento das estratégias e abordagens eficazes na gestão de projetos de desenvolvimento de software.

1.2 Descrição dos capítulos

Este trabalho está dividido em 6 capítulos, apresentado neste capítulo a contextualização e os objetivos. No segundo capítulo é feita uma revisão bibliográfica sobre as metodologias ágeis com foco no Scrum, métricas e ferramentas usadas em gestão. No terceiro capítulo é apresentado a descrição do estudo de caso e suas peculiaridades, adaptações e problemas encontrados. No quarto capítulo é apresentado as métricas desenvolvidas e aplicadas no projeto de estudo. No quinto capítulo é feito uma análise das métricas de acordo com situações reais com o intuito de melhorar e indicar pontos de melhorias, indicando também quais métricas e gráficos podem ser melhorados. Por fim, no sexto capítulo são apresentadas as conclusões e trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

A gestão eficiente de projetos de TI é crucial para o sucesso das organizações. Neste contexto, busca-se cada vez mais eficiência e o sucesso na entrega dos resultados nos projetos de desenvolvimento de software que dependem de práticas adequadas, metodologias ágeis e métricas eficazes. Nesse contexto, o Scrum se destaca como uma das metodologias ágeis mais amplamente adotadas no gerenciamento de projetos de TI. Neste capítulo são apresentadas definições teóricas dos elementos importantes em metodologias ágeis para servir como base para a análise crítica das práticas existentes em um estudo de caso, destacando os benefícios e desafios associados à aplicação desses conceitos.

2.1 Conceitos do Scrum

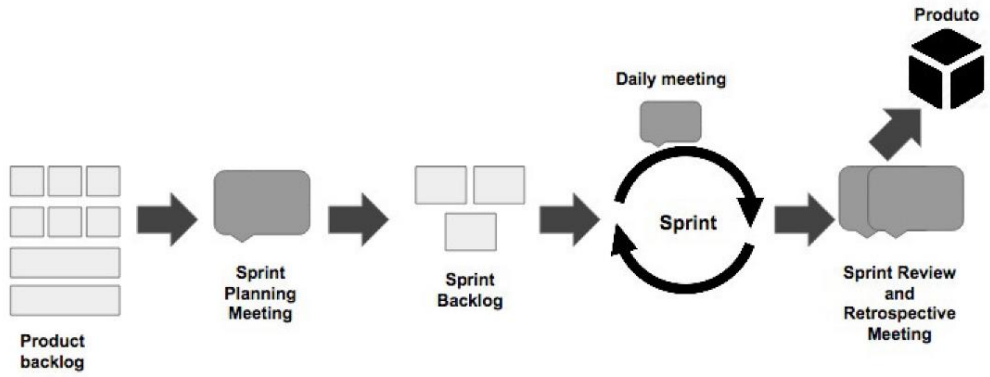
O Scrum teve suas origens no desenvolvimento de software e foi formalizado pela primeira vez em um artigo publicado por Jeff Sutherland e Ken Schwaber em 1995, Schwaber, K., & Sutherland, J. (2020) No entanto, a história do Scrum remonta à década de 1980, quando pesquisadores como Hirotaka Takeuchi e Ikujiro Nonaka começaram a explorar abordagens mais flexíveis para o desenvolvimento de produtos.

Segundo Schwaber e Sutherland (2020), o Scrum se baseia em princípios de transparência, inspeção e adaptação. Ele enfatiza a colaboração entre os membros da equipe, a auto-organização e a comunicação constante com os *stakeholders*.

O Scrum divide o projeto em iterações curtas, chamadas de *Sprints*, que geralmente têm duração de duas a quatro semanas. Em cada *Sprint*, a equipe trabalha para entregar incrementos funcionais e previamente priorizados.

No Scrum, são atribuídos papéis específicos aos membros da equipe, além de eventos e artefatos que são utilizados para facilitar o desenvolvimento de projetos de forma ágil, como podemos ver os conceitos na Figura 1.

Figura 1 – Fluxo Scrum



Fonte: Métricas Ágeis (2017), Casa do código

2.1.1 Papéis

Os papéis específicos dos membros da equipe são:

Product Owner: tem a responsabilidade de definir e priorizar os requisitos do produto. Ele trabalha em colaboração com a equipe de desenvolvimento, garantindo que as necessidades do cliente sejam atendidas e que o produto final seja entregue com qualidade. O *Product Owner* é responsável por maximizar o valor do produto e do trabalho da equipe de desenvolvimento, segundo Schwaber e Sutherland (2020).

Scrum Master: responsável por manter o processo e garantir que as práticas do Scrum sejam seguidas corretamente, removendo obstáculos que possam afetar a equipe de desenvolvimento e facilitando a comunicação entre os membros da equipe. O *Scrum Master* atua como um líder-servidor, ajudando a equipe a entender e implementar o Scrum, bem como a lidar com as complexidades e desafios que surgem durante o desenvolvimento do produto, Schwaber e Sutherland (2020).

Equipe de Desenvolvimento, também conhecido pelo nome em inglês *dev team*: Schwaber e Sutherland (2020) afirmam que a equipe de desenvolvimento é composta por profissionais multifuncionais, que possuem todas as habilidades necessárias para criar um incremento de produto. Ou seja, possuem a responsabilidade de realizar o trabalho necessário para transformar os requisitos do produto em incrementos de valor.

2.1.2 Eventos

Os eventos no Scrum são elementos fundamentais que proporcionam um ritmo e uma estrutura para o trabalho da equipe, promovendo a colaboração, a transparência, o planejamento e a aprendizagem contínua. O eventos do Scrum são:

Sprint Planning: é o evento que marca o início de cada Sprint. Nesse evento, o *Product Owner* apresenta para a equipe de desenvolvimento os itens do *backlog* que serão trabalhados durante a *Sprint* e estabelece o objetivo a ser alcançado. Segundo Schwaber e Sutherland (2020), *Sprint Planning* é uma colaboração entre o *Product Owner* e a equipe de desenvolvimento para definir o trabalho que será realizado durante a S*print*.

Daily Scrum: Schwaber e Sutherland (2020) explicam que o *Daily Scrum* é uma oportunidade para a equipe de desenvolvimento inspecionar seu progresso e adaptar o plano para atingir o objetivo do S*print*. O *Daily Scrum* é uma reunião diária de curta duração, na qual a equipe de desenvolvimento se reúne para sincronizar suas atividades.

Sprint Review: Segundo Cohn (2006), o *Sprint Review* é uma oportunidade para a equipe de desenvolvimento obter *feedback* valioso, tanto do *Product Owner* quanto dos *stakeholders*, com o objetivo de melhorar continuamente o produto. O *Sprint Review* é uma reunião realizada ao final de cada S*print*, na qual a equipe de desenvolvimento apresenta o incremento do produto ao *Product Owner* e aos *stakeholders*.

Sprint Retrospective: Schwaber e Sutherland (2020) destacam que a *Sprint Retrospective* é uma oportunidade para a equipe de desenvolvimento refletir sobre seu trabalho e encontrar maneiras de melhorar continuamente. A *Sprint Retrospective* é uma reunião realizada ao final de cada S*print*, na qual a equipe de desenvolvimento analisa dados e impressões sobre o S*print* anterior. Eles identificam pontos positivos e oportunidades de melhoria, a fim de ajustar seus processos e práticas para futuros *Sprints*.

2.1.3 Artefatos

Product Backlog: segundo Schwaber e Sutherland (2020) afirmam que o *Product Backlog* é a única fonte de requisitos para qualquer alteração a ser feita no produto. O *Product Backlog* é uma lista priorizada de requisitos, funcionalidades e melhorias do produto. É um artefato de responsabilidade do *Product Owner*, que o mantém atualizado e o utiliza para orientar o planejamento dos *Sprints*.

Sprint Backlog: De acordo com Schwaber e Sutherland (2020), o *Sprint Backlog* é um plano emergente que guia o trabalho da equipe de desenvolvimento durante o *Sprint*. O *Sprint Backlog* é uma lista de itens selecionados do *Product Backlog* para serem trabalhados durante o *Sprint* atual. Esses itens são detalhados em tarefas menores e tem seus esforços estimados pela equipe de desenvolvimento durante o *Sprint Planning*.

Incremento: Segundo Cohn (2006), o incremento é uma parte do produto que é entregue ao final de cada *Sprint* e que adiciona valor ao produto como um todo. O Incremento é o resultado do trabalho realizado durante o *Sprint*. É uma versão do produto que está em um estado utilizável e potencialmente entregável, pois passou por testes e atende aos critérios de qualidade estabelecidos.

2.1.4 Histórias de usuário e Story points

As histórias de usuário são utilizadas para representar requisitos de negócio e necessidades dos usuários de forma simples e compreensível. Uma história de usuário descreve uma funcionalidade específica que agrega valor ao usuário final do sistema.

Segundo Cohn (2006), elas capturam o 'quem', 'o quê' e 'por quê' de um requisito, sem entrar em muitos detalhes técnicos. As histórias de usuário são escritas em linguagem natural e seguem uma estrutura simples, geralmente composta por três elementos: uma descrição curta, um critério de aceitação e uma estimativa de esforço.

As histórias de usuário promovem a comunicação eficaz, garantindo que a equipe e os *stakeholders* estejam alinhados em relação aos requisitos e expectativas. Além disso, as

histórias de usuário possibilitam uma abordagem de desenvolvimento iterativa e incremental, com foco no valor do negócio e na entrega de funcionalidades de maneira ágil.

As histórias de usuário são acompanhadas de estimativas de esforço, geralmente em pontos de história (*Story Points*). Essas estimativas ajudam a equipe a planejar o trabalho, definir prazos e lidar com a incerteza ao desenvolvimento de software. Além disso, ao acompanhar o progresso das histórias de usuário ao longo do tempo, é possível obter métricas e dados que contribuem para o projeto.

2.2 Métricas no Scrum

As métricas no Scrum são recursos que auxiliam na avaliação do progresso do projeto, na identificação de possíveis problemas e no monitoramento do desempenho da equipe. Elas fornecem dados objetivos e quantificáveis, permitindo uma análise mais precisa e facilitando a identificação de oportunidades de melhoria. Algumas métricas comumente utilizadas no Scrum incluem:

- Velocidade;
- Velocidade média;
- Eficiência da Sprint.

2.2.1 Velocidade e Velocidade média

A velocidade é uma métrica fundamental na gestão de projetos de TI, especialmente no contexto do *framework Scrum*. No *Scrum*, a velocidade refere-se à capacidade da equipe de desenvolvimento em entregar valor ao longo das iterações (*Sprints*). Ela é calculada medindo-se a quantidade de trabalho concluído pela equipe em um determinado período de tempo.

Segundo Schwaber (2004), a velocidade é uma métrica importante para a previsibilidade e o planejamento do projeto. Ela fornece informações valiosas sobre a produtividade da equipe, ajudando a determinar a quantidade de trabalho que pode ser realizada em cada *Sprint* e a estimar prazos e entregas futuras.

A velocidade é medida em unidades de trabalho, geralmente representadas por pontos de história (*Story Points*) ou unidades de tempo, como horas. Os *Story Points* são uma medida relativa de complexidade, esforço e incerteza envolvidos em uma história do usuário ou uma

tarefa. Eles são atribuídos pela equipe com base em discussões colaborativas e usando-se de técnica de estimativa, como o *Planning Poker*.

Ao longo do tempo, a equipe acumula dados sobre sua velocidade média, que pode ser usada como uma referência para o planejamento do projeto. Por exemplo, se a equipe tem uma velocidade média de 20 *story points* por *Sprint*, é possível estimar que um conjunto de histórias com um total de 40 *story points* provavelmente levará duas *Sprints* para ser concluído.

O cálculo da velocidade média é realizado com base no histórico das *Sprints* anteriores. Para calcular a velocidade média, os seguintes passos podem ser seguidos:

1- Seleção do período (três ou quatro últimas *Sprints*);

2- Somatório dos *story points* das *Sprints* selecionadas;

3- Divisão do total de *story points* pelo número de *Sprints* selecionadas;

4- Obtensão do resultado da velocidade média da equipe.

Calcular a velocidade média no Scrum permite estimar a capacidade de entrega da equipe, planejar o projeto adequadamente e identificar oportunidades de melhoria.

2.2.2 *Eficiência da Sprint*

A eficiência da *Sprint* está relacionada à capacidade da equipe de desenvolvimento de realizar o trabalho planejado dentro do tempo estabelecido para a *Sprint*. Isso ajuda a identificar desvios e tomar ações corretivas durante a *Sprint*, mantendo o projeto alinhado com os objetivos.

O cálculo da eficiência da *Sprint* no Scrum envolve a comparação entre as tarefas planejadas e as efetivamente concluídas durante a Sprint. Para calcular a eficiência, siga os seguintes passos:

1- Tarefas planejadas: Listar todas as tarefas planejadas para a *Sprint*, juntamente com sua estimativa de esforço em horas ou *story points*;

2- Tarefas concluídas: Ao final da *Sprint*, identifique as tarefas que foram completamente concluídas;

3- Cálculo da eficiência: Divida o número de tarefas concluídas pelo número de tarefas planejadas e multiplique por 100 para obter a porcentagem de eficiência da *Sprint*.

Calcular a eficiência da *Sprint* no Scrum é uma prática importante para avaliar o desempenho da equipe e a qualidade das entregas. Essa métrica permite monitorar o progresso, identificar gargalos e promover melhorias contínuas no processo de desenvolvimento de software.

No entanto, é importante lembrar que a eficiência da *Sprint* deve ser analisada em conjunto com outras métricas e práticas ágeis para uma gestão eficaz de projetos de TI.

2.3 *Planning Poker*

Segundo Cohn (2006), o *Planning Poker* é uma técnica colaborativa amplamente utilizada em projetos ágeis, que permite que a equipe estime o esforço necessário para concluir as tarefas através de uma abordagem participativa e baseada no consenso.

Uma técnica colaborativa de estimativa usada em projetos ágeis, como o Scrum. Essa prática envolve a participação da equipe de desenvolvimento e é especialmente útil quando se trata de estimar o esforço necessário para concluir as tarefas do projeto. O objetivo alcançar estimativas mais precisas e realistas por meio da contribuição e do consenso da equipe.

Durante uma sessão de *Planning Poker*, a equipe se reúne para discutir e estimar as tarefas do projeto. Cada tarefa é apresentada e discutida em detalhes para que todos tenham uma compreensão clara dos requisitos e da complexidade envolvida. Em seguida, cada membro da equipe seleciona uma carta de estimativa que representa sua opinião sobre o esforço necessário para concluir a tarefa.

As cartas de estimativa usadas no *Planning Poker* seguem a sequência de Fibonacci. As cartas podem ter valores como 1, 2, 3, 5, 8, 13, etc. Essa sequência é usada para refletir a incerteza inerente às estimativas e evitar a tendência de fornecer estimativas excessivamente precisas. Cada membro da equipe revela sua carta e, se houver divergências nas estimativas, é realizada uma discussão para compreender as diferentes perspectivas.

2.4 Ferramentas de gestão e análise de dados

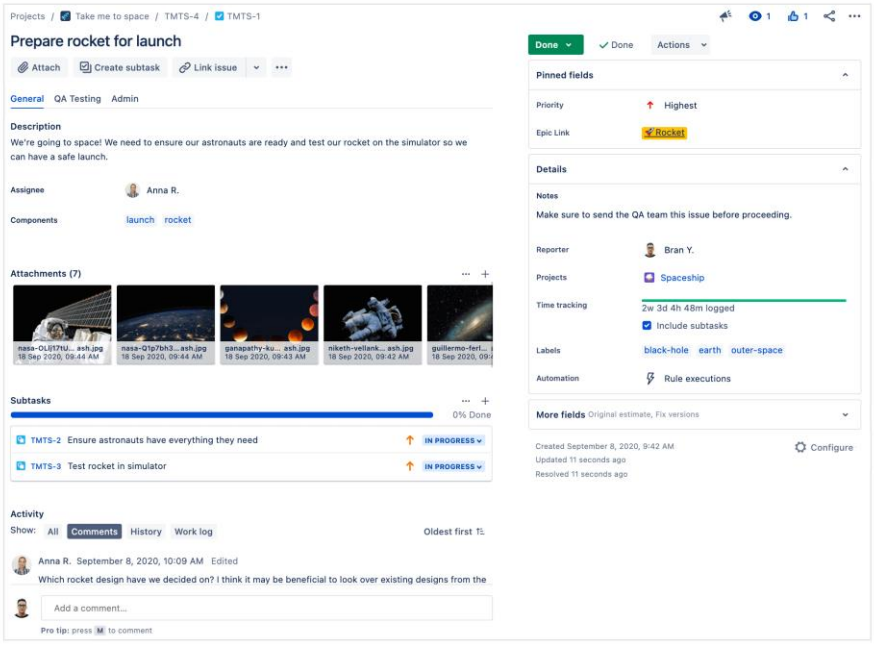
A eficiência das ferramentas de gestão e analisé de dados são essenciais para avaliar o desempenho da equipe e monitorar a entrega de valor ao longo do tempo. Sendo possível identificar oportunidades de melhoria, promover a aprendizagem contínua e aprimorar o planejamento do projeto.

2.4.1 Jira

O Jira é definido como uma ferramenta de gerenciamento de projetos que permite rastrear problemas e colaborar com sua equipe (Guzik, M., 2019, p. 13). Ele fornece recursos para o planejamento, acompanhamento e entrega de projetos, permitindo que as equipes gerenciem tarefas, atribuam responsabilidades, acompanhem o progresso e colaborem em um ambiente centralizado.

Uma das principais vantagens do Jira é a capacidade de rastrear e gerenciar problemas e tarefas de forma eficiente (Guzik, M., 2019, p. 25). Forncecendo recursos para a criação de tickets, atribuição de responsáveis, definição de prioridades, acompanhamento de status e registro de comentários, permitindo um controle detalhado das atividades e um histórico completo de ações realizadas.

Figura 2 - Exemplo de uma história do JIRA



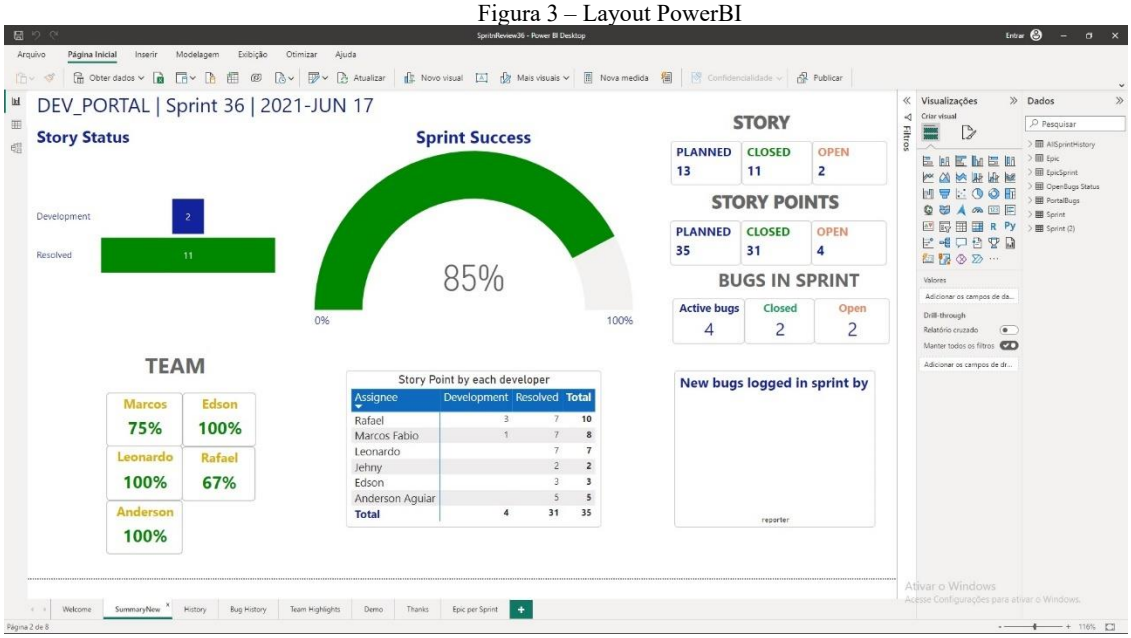
Fonte: Site Atlassian.com

2.4.2 PowerBI

O Power BI é definido como uma ferramenta de análise de negócios da Microsoft que permite o monitoramento de dados em tempo real e tomada de decisões baseadas em *insights* (Moss, B., & Knight, K., 2019). Solidificando-se como uma ferramenta de análise de dados e business intelligence desenvolvida pela Microsoft e que oferece um conjunto de recursos que permitem conectar-se a várias fontes de dados, modelar e transformar os dados, criar visualizações interativas e compartilhar relatórios.

Uma das principais vantagens do Power BI é sua capacidade de fornecer visualizações interativas e personalisáveis dos dados (Rad Reza, 2019, p. 20). Com o Power BI, é possível criar gráficos, Tabelas dinâmicas, mapas e outros tipos de visualizações que permitem explorar os dados de forma intuitiva (Figura 2) e descobrir insights valiosos.

Figura 3 – Layout PowerBI



Fonte: Elaborado pelo autor.

Estudados os fundamentos para análise e, assim, auxiliar no aprimoramento da produtividade da equipe, nos próximos capítulos discutiremos sobre os aspectos do estudo de caso e consquentemente o uso dos conhecimentos para criação das métricas e gráficos.

3 ESTUDO DE CASO: MANUTENÇÃOO E EVOLUÇÃO DE UM PORTAL

Os estudos de caso são uma ferramenta importante na pesquisa acadêmica, permitindo a análise aprofundada de casos reais e a aplicação dos conceitos teóricos em situações práticas. No contexto da gestão de projetos de TI, os estudos de caso fornecem *insights* valiosos sobre a aplicação das métricas e metodologias na prática.

Através de um estudo de caso, analisaremos neste capítulo um exemplo concretos de um projeto de desenvolvimento de software que adotou metodologias ágeis, como o Scrum, e utiliza métricas para gerenciar e avaliar seu desempenho.

3.1 Estrutura do projeto

O caso em estudo se passou em uma equipe que trabalhou no desenvolvimento de um portal de auxílio informativo e de desenvolvimento tecnológico para os funcionários e desenvolvedores internos de uma grande empresa de tecnologia.

Este portal servia como referência técnica para as tecnologias desenvolvidas na empresa, contendo documentos, arquivos e materiais informativos(tutoriais, definições conceituais, guias, documentação contratul a ser seguida etc).

O portal já se encontrava em produção e em uso dos colaboradores. Inicialmente, desenvolvido por uma equipe indiana, o portal foi realocado para outra equipe, sendo esta pertencente a uma empresa brasileira, cujo objetivo era realizar a sua manutenção e adicionar novas funcionalidades, conforme o *feedback* dos usuários e *stakeholders*.

O projeto contava com um time de desenvolvimento formado por desenvolvedores, analistas de qualidade e gerente de projeto. O gerente deste projeto atuava em vários outros projetos e a sua função se concentrava em oferecer suporte administrativo e o tratar com cliente sobre questões contratuais e estruturais.

3.2 Fatores motivadores para mudança da equipe

Um dos fatores relativos ao motivo da substituição da equipe indiana por uma brasileira foi o fato de a equipe anterior possuir um fuso horário diferente, com muitas horas de diferença em relação ao fuso horário do cliente. Estando nos Estados Unidos, a diferença estava impactando as reuniões e o andamento do projeto.

Um outro fator relevante foi a ausência de dados específicos, relacionados ao desenvolvimento do projeto, assim, necessária uma participação mais ativa do cliente no processo de desenvolvimento.

3.3 Adaptações do Scrum

Diferentemente do Scrum, que divide os papéis do time em *Product Owner*, *Scrum Master* e *Development Team*, neste projeto o cliente alocou um funcionário para trabalhar no projeto. Sendo bastante ativo e participando do processo de desenvolvimento, o funcionário alocado pelo cliente atuou simultaneamente como *Product Owner* e *Scrum Master*, o que não é comumente visto e nem recomendado para este método ágil. A justificativa apresentada para esta adaptação metodológica, entretanto, foi o fato de o funcionário possuir vasta experiência no desenvolvimento do portal e vasta experiência acumulada como *Scrum Master*.

Esta adaptação permitiu economia de recursos por parte do cliente, reduzindo o tamanho da equipe e um controle maior na execução do projeto, mas gerou alteração na estrutura que a literatura propõe e consequências que serão abordadas nos tópicos posteriores.

3.3.1 Eventos do Scrum

As adaptações dos eventos do Scrum propostas foram:

Sprint Planning: o funcionário do cliente, exercendo simultaneamente os papéis de *Product Owner* e *Scrum Master*, priorizava e discutia com a equipe de desenvolvimento as funcionalidades do *Backlog* que entrariam no fluxo de desenvolvimento da *Sprint*. O controle do tempo de discussões e *timebox* do evento controlado pelo cliente (*Product Owner* e *Scrum Master*).

Sprint Retrospective: eram discutidas melhorias de processo de desenvolvimento com o cliente, gerente de projeto e a equipe de desenvolvimento. Diante desta participação do cliente, por vezes se fazia necessário uma reunião extra, antes da retrospectiva, com a equipe de

desenvolvimento, analistas de qualidade e gerente de projeto para discutir fatos e melhorias da *Sprint* para a equipe interna sem o cliente, gerando reuniões a mais do que o modelo Scrum prevê.

Sprint Review: onde eram apresentadas pela equipe as entregas da quinzena para o *Product Owner* (cliente) e outros *stakeholders* do cliente. A *Sprint Review* era conduzida por um analista de qualidade que apresentava as métricas geradas com os dados da Sprint e histórico.

Daily Scrum: reuniões diária com o cliente (*Product Owner*/*Scrum Master*) onde o time se mantinha atualizado da situação do trabalho em progresso e de possíveis bloqueios existentes.

3.3.2 *Estimativas e gerenciamento do Backlog*

Diante de uma maior participação e controle sobre o projeto pelo cliente, o processo de estimativas e gerenciamento inicialmente eram feito baseado na antiga equipe de desenvolvimento e não possuía métricas para análise dos resultados de *Sprints* anteriores e posteriores. Assim, foram necessárias adaptações para esta nova equipe brasileira, pois se tratava de uma equipe nova e possuindo um período de aprendizado ate conseguir uma boa produtividade. Logo, a criação de métricas para analisar e melhorar o processo de desenvolvimento se fez necessária.

O gerenciamento do *Backlog* era feito por meio da ferramenta Jira, licenciada e fornecida pelo cliente. A ferramenta foi utilizada para dar celeridade ao processo de desenvolvimento, visto que o Jira permitiu o acesso as histórias feitas anteriormente pela antiga equipe de desenvolvimento e serviu de coleta de dados para criar as métricas necessárias para medir o desempenho e facilitar o gerenciamento através dos recursos existentes (quadros Kanban/Scrum, fluxo de trabalho personalizados, etc). Assim, todas essas informações ajudam a equipe a priorizar trabalho e garantir o cumprimento.

As estimativas das atividades que iriam ser trabalhadas eram feitas na *Planning* com o uso da técnica de *Planning Poker* pela equipe de desenvolvimento.

3.3.3 *Ferramentas utilizadas*

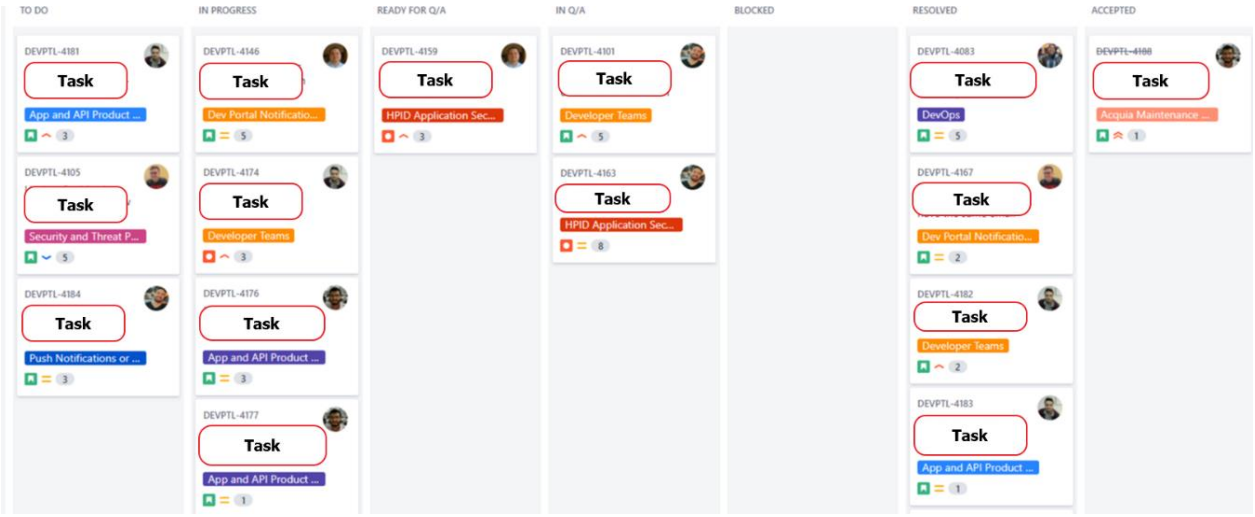
O Jira foi a ferramenta utilizada para fazer o monitoramento das tarefas e acompanhamento do projeto. Através das exportação dos dados, foi possível utilizar outras

ferramentas para análise e construção das métricas. O *Product Owner*, por meio dessa ferramenta, gerenciava e criava novas demandas de atividades (Figura 2), assim como os *bugs* encontrados no sistema.

O fluxo de trabalho (Figura 4) em uma *Sprint* esta dividido nas seguintes etapas:

- *To Do*: etapa na qual as histórias ou bugs que estavam refinadas e priorizadas entravam para o fluxo de desenvolvimento do time.
- *In Progress*: etapa na qual os desenvolvedores estão trabalhando no desenvolvimento das atividades.
- *Blocked*: atividades que possuem algum tipo de bloqueio ou impedimento.
- *Ready for Q/A*: etapa em que os desenvolvedores disponibilizam as atividades para serem testadas no ambiente de desenvolvimento e homologação.
- *In Q/A*: etapa em que são testadas as funcionalidades no ambiente de desenvolvimento e homologação.
- *Resolved*/*Accepted*: etapa onde o *Product Owner* valida se as funcionalidades atingiam o objetivo definido para o negócio. Isto ocorre após a validação da equipe de Qualidade em que apos os teste validados eram movidas as histórias/*bugs* para a coluna *Resolved*. Na *Sprint Review*, o *Product Owner*/*Scrum Master* validava as histórias/bugs que estavam na coluna *Resolved* e movia para a coluna Accepted.

Figura 4 - Quadro do fluxo de trabalho



Fonte: Elaborado pelo autor.

3.4 Problemas encontrados no projeto e prospecção de melhorias

3.4.1 *Estimativa de histórias*

Diversos fatores podem interferir na produtividade da equipe de desenvolvimento. Entre eles, encontraram-se alguns problemas relacionados a estimativas de histórias (dimensionadas em *Story Points*) por cada desenvolvedor em cada ciclo da *Sprint*, ocasionada, neste contexto, por se tratar de uma equipe nova, um projeto que já estava em produção e um sistema em processo de manutenção e melhoria. Com a ausência de métricas usadas em *Sprints* anteriores, tornou-se uma tarefa difícil estimar as atividades durante a *Sprint Planning* e, posteriormente, analisar o desempenho da equipe para melhorar o processo de desenvolvimento.

Logo, decidiu-se criar métricas que auxiliassem o processo de desenvolvimento e pudessem auxiliar no planejamento de tarefas e na capacidade produtiva da equipe de desenvolvimento, sem causar frustrações por não cumprimento das atividades da *Sprint*. No próximo capítulo, serão detalhadas as métricas criadas.

3.4.2 *Acúmulo de papéis*

Outro problema encontrado neste projeto é o acúmulo de papéis por parte do *Scrum Master* e *Product Owner*. Como aponta Schwaber e Sutherland (2020), o *Scrum Master* é responsável por garantir que o Scrum seja compreendido e implementado corretamente, facilitar as reuniões e remover obstáculos que possam impedir o progresso da equipe. No entanto, o cliente possuía o papel de *Scrum Mast*er e *Product Owner*, o que prejudica a distribuição adequada das responsabilidades.

O acúmulo de papéis por parte do *Scrum Master* resultou em uma sobrecarga de trabalho e falta de tempo para se concentrar nas atividades essenciais. Além disso, a equipe se tornou dependente do *Scrum Master/Product Owner* para tomar decisões importantes ou resolver problemas, o que comprometeu a autonomia e a capacidade de autogerenciamento do time.

Devido a essa sobrecarga de trabalho, muitas vezes as histórias não estavam bem escritas ou bem definidas e necessário durante a *Sprint Planning* ser ajustada ou reescritas. Resultando no aumento do tempo das reuniões.

Diante dessa participação ativa do cliente no eventos relacionados ao projeto, houve a necessidade de reuniões entre a equipe de desenvolvimento e o gerente do projeto para alinhar questões que não poderiam ser tratadas com o cliente participando e, assim, gerando mais reuniões do que as previstas pelo Scrum.

4 Métricas desenvolvidas

Com o uso da exportação de dados do JIRA, foi possível filtrar e gerar arquivos no final de cada *Sprint* com os dados da *Sprint* atual, histórico das *Sprint* e os *bugs* do portal.

Utilizando o software PowerBI, essas planilhas foram manipuladas e, em seguida, foi gerado um relatório com todos os gráficos e métricas, discutidas nas próximas seções, para serem apresentados ao cliente e *stakeholders* durante a *Sprint Review*.

As métricas e gráficos foram desenvolvidas conforme lista abaixo:

- Sprint Atual
    - *Story Status por Sprint*
    - *Sprint Sucess*
    - *Story, Story Points* e *Bugs in Sprint*
    - *Team*
    - *Story Point by each developer*
- Histórico das Sprints
    - *Sprint Success Percentage History*
    - *Sprint Velocity History*
    - *Sprint Success History*
    - *Open Bugs History*
- Bugs do Portal
    - *Open Bugs Priority*
    - *All Bugs Status*

4.1 *Sprint* Atual

Tabela 1 - Tabela com os dados da Sprint Atual

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	key	title	summary	link	type	priority	status	assignee	email	Story point	reporter	created	
2	DEVPTL-4165	[DEVPTL-4165]	CLONE - Regre	https://jira.cso-h	Story	Medium	Resolved	Anderson Aguia	anderson.aguiar	2	David Schneider	Wed 2 Jun 2021 20:44:06 +0000	
3	DEVPTL-4164	[DEVPTL-4164]	Report on ALL o	https://jira.cso-h	Story	Medium	Resolved	Anderson Aguia	anderson.aguiar	3	David Schneider	Wed 2 Jun 2021 20:41:23 +0000	
4	DEVPTL-4163	[DEVPTL-4163]	Some app expire	https://jira.cso-h	Bug	Medium	Development	Marcos Fabio	marcos.fabio@h	1	Linda Kingham	Wed 2 Jun 2021 19:29:18 +0000	
5	DEVPTL-4162	[DEVPTL-4162]	Test the ability to	https://jira.cso-h	Story	High	Resolved	William Jehnyffe	william.jehnyffer	2	David Schneider	Tue 1 Jun 2021 18:55:58 +0000	
6	DEVPTL-4161	[DEVPTL-4161]	Investigate how	https://jira.cso-h	Story	Medium	Resolved	Marcos Fabio	marcos.fabio@h	2	David Schneider	Tue 1 Jun 2021 18:32:32 +0000	
7	DEVPTL-4160	[DEVPTL-4160]	Alternate client s	https://jira.cso-h	Bug	Medium	Resolved	Rafael Marques	rafael.marques@	2	David Schneider	Tue 1 Jun 2021 18:29:58 +0000	
8	DEVPTL-4159	[DEVPTL-4159]	HP ID App's sec	https://jira.cso-h	Bug	High	Development	Rafael Marques	rafael.marques@	3	David Schneider	Tue 1 Jun 2021 18:24:16 +0000	
9	DEVPTL-4156	[DEVPTL-4156]	Edit App Page -	https://jira.cso-h	Story	Medium	Resolved	Marcos Fabio	marcos.fabio@h	2	David Schneider	Mon 24 May 2021 18:33:08 +0000	
10	DEVPTL-4152	[DEVPTL-4152]	Update validatio	https://jira.cso-h	Story	High	Resolved	Marcos Fabio	marcos.fabio@h	3	David Schneider	Thu 20 May 2021 19:08:44 +0000	
11	DEVPTL-4140	[DEVPTL-4140]	E-mail notificatio	https://jira.cso-h	Story	High	Resolved	Rafael Marques	rafael.marques@	5	David Schneider	Tue 18 May 2021 17:01:02 +0000	
12	DEVPTL-4136	[DEVPTL-4136]	Apigee Connecti	https://jira.cso-h	Story	High	Resolved	Leonardo Monte	leonardo.monter	2	David Schneider	Wed 12 May 2021 00:03:31 +0000	
13	DEVPTL-4129	[DEVPTL-4129]	Update Apigee (	https://jira.cso-h	Story	High	Resolved	Leonardo Monte	leonardo.monter	5	David Schneider	Tue 4 May 2021 19:44:35 +0000	
14	DEVPTL-2297	[DEVPTL-2297]	"Add Contact" se	https://jira.cso-h	Bug	High	Resolved	Edson Brito	edson.brito@hp.	3	Laxmitulasi Thal	Mon 28 Jan 2019 13:12:18 +0000	
15													

Fonte: Elaborado pelo autor.

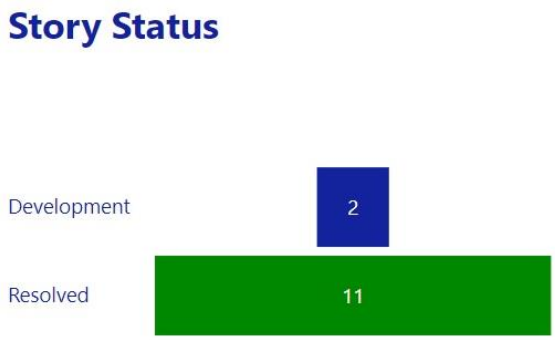
Na tabela 1, as colunas representam os valores obtidos da exportação dos dados do Jira e através delas se criou-se os gráficos e métricas das seções posteriores.

4.1.1 *Story Status por Sprint*

Indica o quantitativo das atividades em cada estado por *Sprint*, de acordo com a coluna "status" do arquivo gerado (Tabela 1). O referente "status" das atividades é conFigurado no Jira de acordo com o quadro do fluxo de trabalho. A Figura 5 mostra como é apresentado aos *stakeholders* o gráfico de barras que permite visualizar a proporção de histórias concluídas e em desenvolvimento na *Sprint*.

$$Story\_Status = \sum num\_hist\_e\_bugs\_status$$

Figura 5 - representação do Calculo de Story points por status



Fonte: Elaborado pelo autor.

4.1.2 *Sprint Sucess*

Indica o cálculo percentual do somatório de histórias/*bugs* que possuem o "status" equivalente a "*resolved*" ou "*accepted*" da Tabela Sprint atual (Tabela 1), que é apresentada na Figura 6. Determinando assim, o quantitativo percentual referente a conclusão das atividades planejadas na Sprint em analise.

$$\%Sprint\_Sucess = \frac{\sum hist\_bug\_resolved + \sum hist\_bugs\_accepted}{\sum total\_hist\_bugs} \times 100$$

Figura 6 - Gráfico de representação de cálculo do sucesso da Sprint

Fonte: Elaborado pelo autor.

Segundo Cohn (2006), uma boa prática de monitorar a eficiência da *Sprint* é analisar se a velocidade real está se desviando da velocidade planejada. Logo, o ideal para essa métrica é calcular a percentagem de acordo com o número de *story points* das histórias completas pelo número de *story points* planejados no início da *Sprint*. Logo, a análise e alteração desse cálculo serão apresentadas no capítulo 5.

4.1.3 *Story, Story Points* e *Bugs in Sprint*

Representação do somatório das histórias, pontos de histórias e os *Bugs* da *Sprint* (este solicitado para elucidar o quantitativo de *bugs* trabalhados na *Sprint*) que foram planejadas, fechadas (concluídas) e continuam abertas (não concluídas).

Na Figura 7 tem-se um exemplo de como são apresentados aos stakeholders os elementos quantitativos do somatório, conforme as *Story, Story Points e Bugs*.

Para representação de *Story, Story Points e Bugs*, o numero de linhas do arquivo gerado (Tabela 1) representa a quantidade de *Story* que foram trabalhadas na *Sprint* em analise. Assim, pode-se criar as métricas abaixos para serem inseridas na Figura 7.

Figura 7 - Representação do calculo das histórias, story points e bugs

Fonte: Elaborado pelo autor.

$$Story\_Planned = \sum historias\_Sprint$$

$$Story\_Closed = \sum historias\_Resolved + \sum historias\_Accepted$$

$$Story\_Open = Story\_Planned - Story\_Closed$$

Para representação dos *Story Points*, usam-se as colunas *status* e *story point* do arquivo gerado (Tabela 1). Conforme as métricas definidas abaixo, foram calculados e inseridos na Figura 7.

$$SP\_Planned = \sum SP\_historias\_Sprint$$

$$SP\_Closed = \sum SP\_historias\_\text{Resolved} + \sum SP\_historias\_Accepted$$

$$SP\_Open = SP\_Planned - SP\_Closed$$

Para a representação dos *Bugs in Sprint*, usam-se também a coluna *status* e a quantidade de linhas em que o valor da coluna *type* possui o valor igual a *Bug* do arquivo gerado (Tabela 1).

$$Bugs\_Planned = \sum bugs\_Sprint$$

$$Bugs\_Closed = \sum bug\_\text{Resolved} + \sum bugs\_\text{Closed}$$

$$Bugs\ Open = Bugs\_Planned - Bugs\_Closed$$

A contabilização de *Stories*, *Story Points* e *Bugs* em uma *Sprint* permite que o *Scrum Master* analise (em conjunto com as outras métricas) se o planejamento das atividades da *Sprint* precisa ser revisto/ajustado ou se algum fator externo impactou na produtividade. Esses dados são discutidos e avaliados durante a reunião de retrospectiva da *Sprint*.

4.1.4 *Team*

Percentual de conclusão das atividades de cada membro do time, de acordo com as atividades que possuem o "status" equivalente a "resolved" ou "accepted" do arquivo gerado (Tabela 1). Para cada membro da equipe temos:

$$\%Assigned\_por\_Membro = \frac{\sum historias\_assigned\_\text{resolved} + \sum historias\_assigned\_\text{accepted}}{\sum historias\_assigned} \times 100$$

Figura 8 - Representação da Percentagem de conclusão das atividades por membro da equipe

Fonte: Elaborado pelo autor.

A representação percentual de cada membro da equipe (Figura 8) permite ao *Scrum Master* acompanhar e analisar o desempenho por membro. Para o gestor do projeto, permitia acompanhar e prestar o suporte ao membro da equipe que possuísse produção baixa.

4.1.5 *Story Point by each developer*

Tabela demonstrativa do quantitativo dos *story points* de acordo com o *status* e por membro da equipe.

Tabela 2 – Representação do numero de story points por membro da equipe e status

Story Point by each developer

Assignee	Development	Resolved	Total
Rafael	3	7	**10**
Marcos Fabio	1	7	**8**
Leonardo		7	**7**
Jehny		2	**2**
Edson		3	**3**
Anderson Aguiar		5	**5**
Total	**4**	**31**	**35**

Fonte: Elaborado pelo autor.

Analisando a Tabela 2 em conjunto com a representação da Figura 8, é possível verificar com mais exatidão o rendimento do membro da equipe, pois, a atividade poderia estar com status igual a *blocked* (fator externo que impedia o desenvolvimento ou dependência de outras equipes) e impactar no cálculo.

4.2 Histórico das *Sprints*

Tabela 3 – Planilha com os dados relevantes de todas as Sprints.

Sprint	Story Delivered	Planned Velocit	Achieved Velocity	Bugs	Story Planned	Bugs Logged o	Closed	Open	SprintSuccess	OpenBugs	Bugs on Sprint
2020-22-Nov19	19	50	46	835	21	1	4	1	90	94	5
2020-23-Dec03	18	37	37	837	18	1	7	0	100	90	7
2020-24-Dec17	9	20	18	839	10	2	3	0	90	89	3
2021-25-Jan14	4	9	9	846	4	2	3	0	100	92	3
2021-26-Jan28	7	15	15	849	7	0	1	0	100	94	1
2021-27-Feb11	6	37	19	849	15	1	3	2	43	92	5
2021-28-Feb25	6	23	12	851	11	1	0	3	55	93	3
2021-29-Mar11	11	35	30	854	12	1	4	0	92	88	4
2021-30-Mar25	14	28	26	855	15	0	2	0	93	85	2
2021-31-Apr07	13	32	32	858	13	1	4	0	100	82	4
2021-32-Apr22	10	29	24	858	11	0	2	0	91	78	2
2021-33-May06	12	34	34	860	12	0	1	2	100	76	1
2021-34-May-2	12	34	29	861	14	0	2	0	86	68	2
2021-35-Jun-0	11	33	33	864	11	0	2	0	100	63	2
This Sprint	11	35	31	865	13	0	2	2	85	60	4

Fonte: Elaborado pelo autor.

Com os dados do histórico das *Sprints* (Tabela 3) e as métricas da *Sprint* (4.1), foi possível criar gráficos demonstrativos com as análises vistas a seguir. A formatação referente a *Sprint*(coluna *Sprint* da Tabela 3) segue o seguinte formato:

Ano - Número da Sprint – dia e mês de início

Como exemplo, o valor igual a 2022-22-Nov19 representa a Sprint 22 que possuiu início em 19 de novembro de 2022.

4.2.1 *Sprint Success Percentage History*

Representação do gráfico (Gráfico 1) com a no eixo X da data/numeração de cada Sprint (coluna “Sprint” da Tabela 3) e no eixo Y o cálculo percentual de conclusão de cada Sprint (coluna “Sprint Sucess” da Tabela 3).

Gráfico 1 – Histórico do sucesso das Sprints

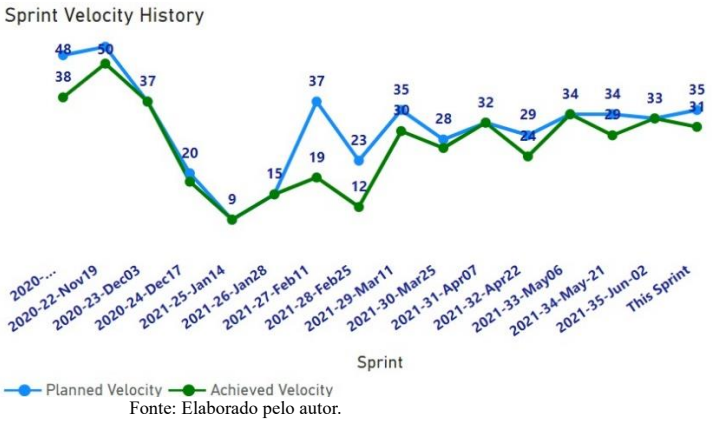
Fonte: Elaborado pelo autor.

Analisando o Gráfico 1, podemos observar o percentual de sucesso no decorrer das *Sprints,* este derivado do cálculo da métrica Sprint Sucess(4.1.1). Assim, o *Scrum Master* pode realizar alguma ação rápida de melhoria no processo ou acompanhar alguma implementação de mudança. No capitulo 6 são analisadas situações exemplo e os impactos desse gráfico, propondo-se uma melhoria.

4.2.2 *Sprint Velocity History*

A representação do Gráfico 2 com o eixo X da data/numeração de cada Sprint (coluna “Sprint” da Tabela 3) e, no eixo Y, o cálculo da velocidade das atividades planejadas (coluna “Planned Velocity” da Tabela 3), além de velocidade das atividades concluídas de cada Sprint (coluna “Archived Velocity” da Tabela 3).

Gráfico 2 – Historico das velocidades planejadas e alcancadas



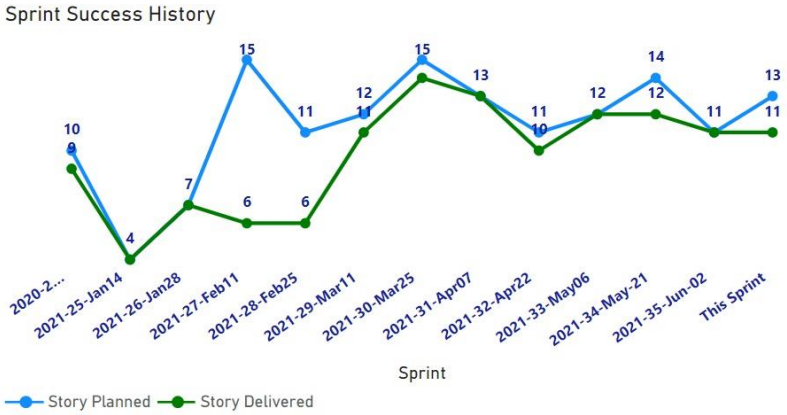
Fonte: Elaborado pelo autor.

Na Gráfico 2, a linha azul representa a quantidade de *story points* planejadas ao inicio da *Sprint.* A linha verde representa a quantidade de *story points* concluidos. Se a distancia entre os ponto e linhas forem maior, isso pode indicar algum problema de dimensionamento ou algum fator que tenha impactado a produtividade da equipe, permitindo ao *Product Owner* ou *Scrum Master* tomarem decisões e promoverem ações para melhorar a produtividade.

4.2.3 *Sprint Success History*

A representação do Gráfico 3 com o eixo X da data/numeração de cada Sprint (coluna “Sprint” da Figura 11) e, no eixo Y, o quantitativo das histórias planejadas (coluna “Planned Velocity” da Figura 11), além de histórias concluídas de cada Sprint (coluna “Achieved Velocity" da Figura 11).

Gráfico 3 – Histórico das histórias planejadas e alcançadas



Fonte: Elaborado pelo autor.

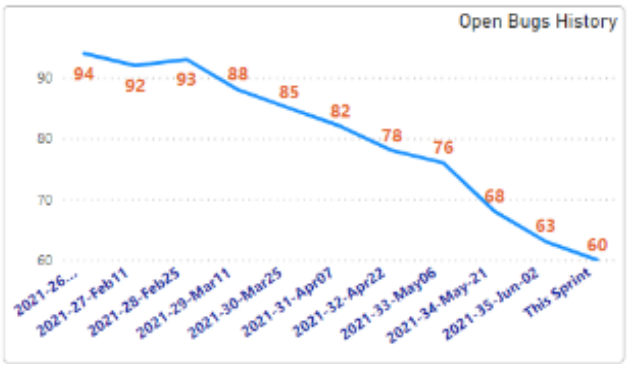
Similar ao *Sprint Velocity History* (Gráfico 2), mas, no Gráfico 3, a linha azul representa a quantidade de histórias planejadas ao inicio da *Sprint* e a linha verde representa a quantidade de histórias concluídas. Isso auxilia à tomada de decisão do *Product Owner* ou *Scrum Master* quando analisado em conjunto com a *Sprint Velocity History.* Isso se explica porque a complexidade da tarefa pode interferir na diferença entre os pontos.

4.2.4 *Open Bugs History*

Gráfico com a representação no eixo X da data/numeração de cada Sprint (coluna "Sprint" da Tabela 3) e, no eixo Y, do somatório de *bugs* abertos na Sprint (valor da coluna "OpenBugs" da Tabela 3)

$$Bugs\_Open\_Sprint = \sum total\_bugs - (\sum bugs\_Resolved + \sum bugs\_Accepted)$$

Gráfico 4 – Histórico dos bugs abertos



Fonte: Elaborado pelo autor.

Esta representação gráfica (Gráfico 4) foi solicitada pelo cliente para fazer um acompanhamento quantitativo de quantos *Bugs* existiam no *Backlog* e para poder aplicar ações para diminui-los conforme suas características e prioridades, visto que o cliente era também *Product Owner* e *Scrum Master.*

4.3 *Bugs* do Portal

Tabela 4 – Dados de todos os Bugs do Backlog

	A	B	C	D	E	F	G	H	I	J	K	L
1	key	title	summary	link	type	priority	status	assignee	email	reporter	created	
2	DEVPTL-3583	[DEVPTL-3583]	On App Revoke	https://jira.cso-hy	Bug	High	IceBox	Unassigned	-1	Ajit Kankanawac	Thu 9 Apr 2020 06:42:58 +0000	
3	DEVPTL-2297	[DEVPTL-2297]	"Add Contact" sa	https://jira.cso-hy	Bug	High	Closed	Anderson Aguiai	anderson.aguiar	Laxmitulasi Thal	Mon 28 Jan 2019 13:12:18 +0000	
4	DEVPTL-4174	[DEVPTL-4174]	Problemas found	https://jira.cso-hy	Bug	High	IceBox	Leonardo Monte	leonardo.monter	Milliam Jehnyffe	Mon 14 Jun 2021 20:41:29 +0000	
5	DEVPTL-4163	[DEVPTL-4163]	Some app expira	https://jira.cso-hy	Bug	Medium	Development	Marcos Fabio	marcos.fabio@h	Linda Kingham	Wed 2 Jun 2021 19:29:18 +0000	
6	DEVPTL-4160	[DEVPTL-4160]	Alternate client s	https://jira.cso-hy	Bug	Medium	Closed	Rafael Marques	rafael.marques@	David Schneider	Tue 1 Jun 2021 18:29:58 +0000	
7	DEVPTL-3306	[DEVPTL-3306]	No confirmation	https://jira.cso-hy	Bug	Medium	IceBox	Unassigned	-1	Ajit Kankanawac	Wed 13 Nov 2019 11:35:21 +0000	
8	DEVPTL-3307	[DEVPTL-3307]	UI and Text issu	https://jira.cso-hy	Bug	Highest	IceBox	Unassigned	-1	Ajit Kankanawac	Wed 13 Nov 2019 11:39:23 +0000	
9	DEVPTL-3305	[DEVPTL-3305]	"Page not found"	https://jira.cso-hy	Bug	Highest	IceBox	Unassigned	-1	Ajit Kankanawac	Wed 13 Nov 2019 11:34:12 +0000	
10	DEVPTL-3304	[DEVPTL-3304]	CRON is not abl	https://jira.cso-hy	Bug	High	IceBox	Unassigned	-1	Ajit Kankanawac	Wed 13 Nov 2019 11:32:21 +0000	
11	DEVPTL-3303	[DEVPTL-3303]	New Articles put	https://jira.cso-hy	Bug	Highest	IceBox	Unassigned	-1	Ajit Kankanawac	Wed 13 Nov 2019 11:29:53 +0000	

Fonte: Elaborado pelo autor.

Com os dados do filtro relacionado aos *bugs* do portal (Tabela 4), foi possível consolidar dados para avaliar o desempenho da equipe em relação ao registro e à correção *bugs*, que são discutidos a seguir.

No sistema do Jira em estudo, as histórias ou bugs em que não estavam alocadas a nenhuma *Sprint* e consequentemente não estavam em desenvolvimento possuíam o status *IceBox*

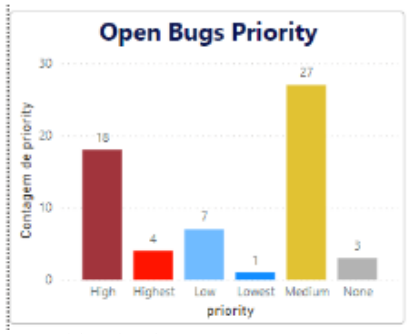
As histórias ou bugs que ja haviam sido concluídas em Sprint anteriores recebiam o status de *Closed*.

4.3.1 *Open Bugs Priority*

Gráfico com a representação no eixo X da prioridade das atividades (coluna *priority* da Tabela 4) e, no eixo Y, o somatório dos *bugs* abertos por cada tipo de prioridade (coluna *type* igual a *Bug* da Tabela 4).

$$Open\_Bugs\_Priority = (Priority)(\sum total\_bugs - (\sum bugs\_Resolved + \sum bugs\_Closed))$$

Gráfico 5 – : Quantidade de *bugs* abertos por prioridade do *Backlog*



Fonte: Elaborado pelo autor.

As representações gráficas *Open Bugs Priority* e *All Bugs Status* (Gráfico 5) ilustram o quantitativo de *bugs* por status, consequentemente usadas para auxiliar na tomada de decisões pelo *Product Owner* de quais *bugs* poderiam ser priorizados a serem trabalhados ou enviados para outra equipes, caso houvesse alguma dependência.

4.3.2 *All Bugs Status*

Tabela 5 representa o status dos *bugs* presentes no *backlog* (coluna "status") e o somatório por cada status.

Tabela 5 – Somatorio dos bugs por status do Backlog

All Bugs Status

status	Contagem de status
Closed	805
IceBox	55
Development	3
Blocked	2
Total	**865**

Fonte: Elaborado pelo autor

Após a criação das métricas apresentadas na seção 4 foi utilizado com o passar das Sprints e buscando a melhoria contínua, houve a necessidade de se analisar a vantagem e a melhoria(modificação) de algumas métricas e gráficos. Conforme mostrado na análise da seção do capítulo seguinte(seção 5).

5 Análise e melhorias das métricas

Utilizado como exemplo as *Sprints* 29 e 27 para elucidar a diferença que pode existir entre o cálculo da métrica de sucesso da *Sprint*, baseando-se no número de *Story*, e o cálculo da métrica de sucesso da *Sprint*, baseando-se Story points obtemos os seguintes valores:

Tabela 6 – Exemplo do calculo de sucesso baseado em Story e Story points

Sprint	Story planned	Story Delivered	%
29	12	11	92
27	15	6	40

Sprint	Planned Velocity	Achived Velocity	%
29	35	30	86
27	37	19	51

Fonte: Elaborado pelo autor

Podemos observar na Tabela 6 que ha uma variação percentual para menos ou para mais se compararmos os cálculos do sucesso da *Sprint*.

Para *Sprint* 27, temos que o cálculo de sucesso baseado em numero de *Story* é igual a 40% e que o cálculo baseado em *story points* (*Velocity*) é igual a 51%.

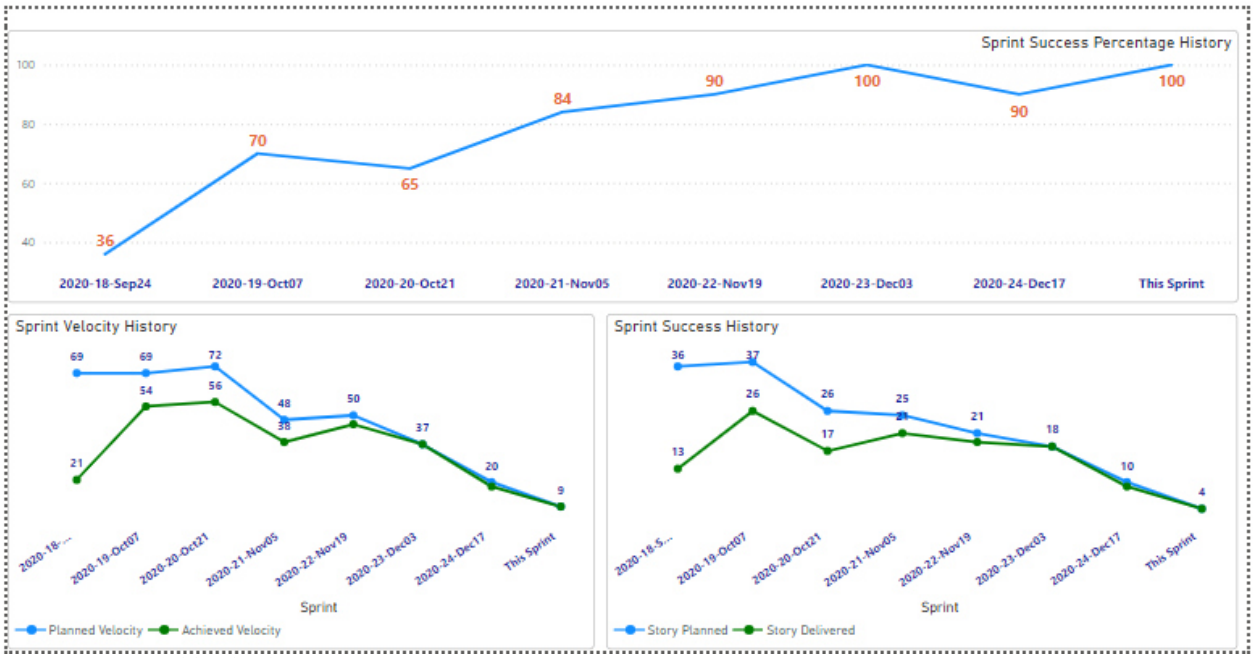
Para *Sprint* 29, temos que o cálculo de sucesso baseado em número de histórias é igual a 92% e para o cálculo baseado em *story points* (*Velocity*) é igual a 86%.

Visando o aprimoramento da métrica de sucesso da Sprint e para deixá-la de acordo com o que a literatura sugere (Cohn 2006), o cálculo de sucesso da *Sprint* deverá ser feito de acordo com o numero de *story points*, ou seja, de acordo com a velocidade da equipe. Mas, como podemos observar neste exemplos práticos, cabe ser discutido a téoria desse uso confome a literatura.

5.1 Métricas relativas ao histórico das Sprints

Com o decorrer das *Sprints*, a planilha referente ao histórico pode ser alimentada e, assim, gerados dados suficientes para uma análise minuciosa.

Gráfico 6 – Historico da Sprint 18 a 23



Fonte: Elaborado pelo autor.

Para melhor compreensão, a análise será feita em situações reais ocorridas no projeto em estudo.

Caso 1: Dimensionamento dos *story points* na equipe nova

Como pode ser visto no Gráfico 6, a *Sprint* de número 18, primeira realizada pela nova equipe brasileira, foi planejada para o desenvolvimento de 36 histórias e somando um total de 69 *story points*. Ao final desta *Sprint*, usaram-se as métricas definidas como forma de análise do trabalho da equipe e observou-se que o sucesso da *Sprint* foi de 36%; *Sprint Velocity* alcançada foi de 21, para 13 histórias entregues. Consequentemente, podemos concluir que o número de histórias para cada desenvolvedor foi superestimado (atribuídas mais atividades do que a equipe podia produzir).

Durante a *Sprint Retrospective*, também foi observado com a equipe que os acessos, configurações de acesso e conFigurações de ambiente de desenvolvimento impactaram na baixa

produtividade da equipe, apesar de estes procedimentos serem inerentes a qualquer equipe de desenvolvimento de software, mas que deveriam ter sido levadas em consideração na *Sprint Planning*.

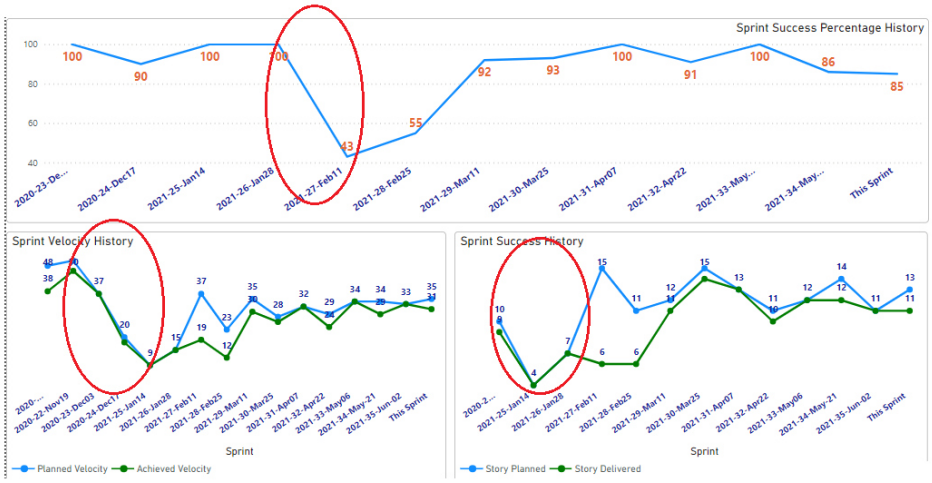
Na *Sprint* 19, foram mantidos valores próximo aos anteriores para histórias e *story points*, com o objetivo de analisar a produtividade para uma iteração em que situações empíricas não estivessem presentes e impactaram na *Sprint* anterior. Logo, percebeu-se que *Sprint Success* da Sprint 19 (70%) foi superior ao da Sprint anterior (36%), concluídos 54 story points (69 planejados) e 26 histórias entregues (37 planejadas).

Na Sprint 20, foram mantidas previsões próximas às anteriores para o número de histórias e *story points*. Buscava-se avaliar se a equipe estava em processo de adaptação, visto que, para equipes novas, existe um período para estabilização e aquisição de conhecimento a respeito dos detalhes técnicos do projeto e de suas regras de negócio. Registrou-se o desempenho de 65% (*Sprint Sucsess*), com 17 histórias entregues, mas também um pequeno aumento de 56 *story points* concluídos em relação ao número de *Story Points* produzidos na iteração anterior (que foi de 54 *Story Points*).

Nas Sprints 21 a 23, O *Product Owner/Scrum Master* adaptou o número de histórias trabalhadas na *Sprint* com o intuito de conseguir valores para calcular um valor aproximado médio ideal para cada *Sprint*. Assim, podemos observar que os valores referentes ao *Sprint Success* subiu (84% e 90%) e a equipe encontrou um valor médio em que obtivesse uma boa produtividade.

Caso 2: Mudanças estruturais na equipe

Gráfico 7 – Historico da Sprint 23 a 35



Fonte: Elaborado pelo autor

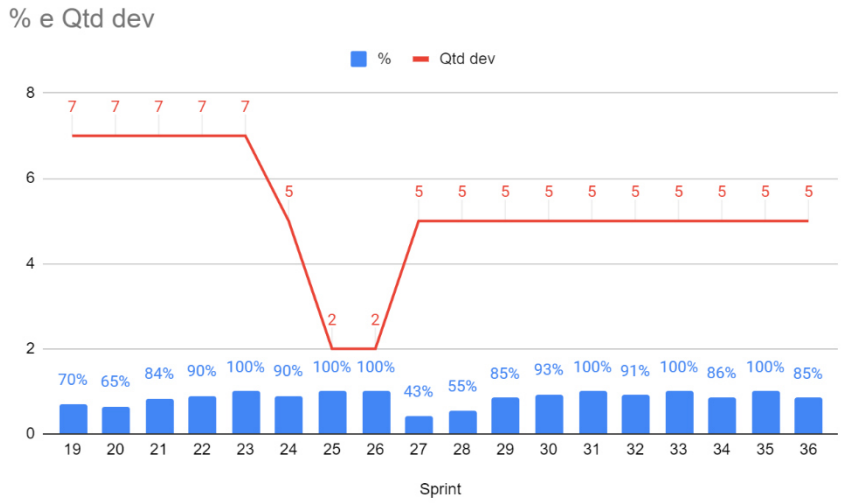
Durante a Sprint 23(Gráfico 7), uma decisão de negócio fez a equipe se dividir e trabalhar no desenvolvimento de um outro portal interno. Isso reduziu a quantidade de atividades trabalhadas neste projeto, necessária a contratação de novos membros.

Se analisarmos a Sprint 24 e 25, nota-se uma queda abrupta na velocidade e no número de histórias entregues pelos gráficos *Sprint Velocity History* e *Sprint Success History*. Entretanto, não conseguimos detectar essa mesma queda no gráfico *Sprint Success Percentage History*, pois para Sprint 24 e 25 foram registrados os valores de 90% e 100%. Se fosse analisado somente a métrica *Sprint Success Percentage History,* não seria percebida essa queda abrupta na velocidade e no número de histórias entregues. Ou seja, observa-se a importância de analisar os gráficos em conjunto.

Como forma de melhorar os gráficos e poder observar essa situação (caso 2) de forma rápida e prática, propõe-se adicionar aos gráficos existentes de *Sprint Success Percentage*, *Sprint Velocity History* e *Sprint Success History* o número de membros da equipe em cada Sprint. Isto se justifica, pois, sem o entendimentos dos fatos ocorridos no projeto, não se consegue compreender as oscilações pela análise exclusiva das métricas.

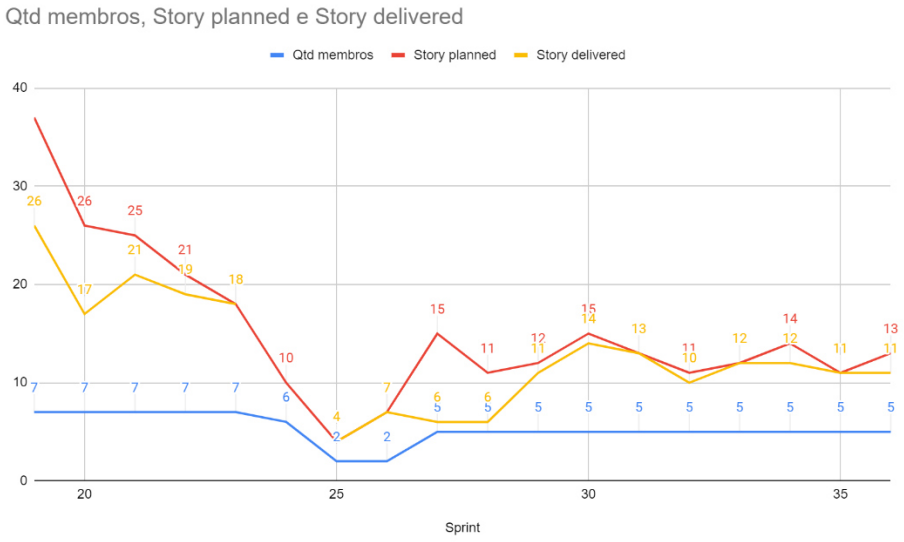
Propõe-se adicionar às métricas e aos gráficos existentes o número de membros da equipe de desenvolvimento que trabalho em cada Sprint. Logo, para calcular a quantidade de membros por equipe, será necessário usar a planilha com os dados da *Sprint* atual e somar a quantidade de linhas distintas (sem repetir o numero de membros) presentes na coluna *assigne*.

Gráfico 8 – Sprint Success History com quantidade de membros da equipe



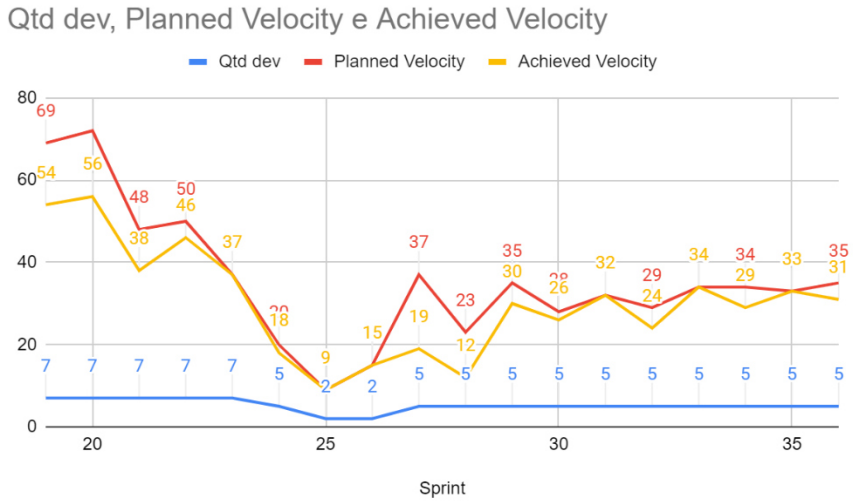
Fonte: Elaborado pelo autor

Gráfico 9 – Sprint Story History com quantidade de membros da equipe



Fonte: Elaborado pelo autor

Gráfico 10 – Sprint Velocity History com quantidade de membros da equipe



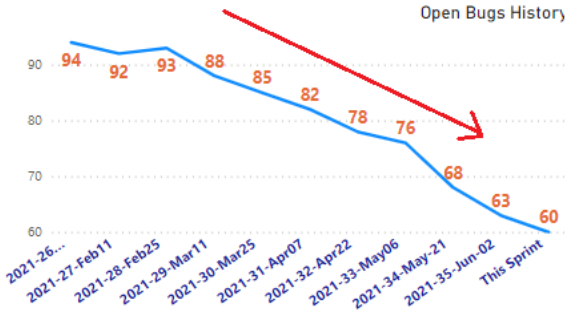
Fonte: Elaborado pelo autor

5.2 Métricas relativas aos *Bugs*

Com o uso da planilha dos *bugs* do portal, podemos ter um panorama rápido e visual dos *bugs* presentes no *backlog*.

No gráfico de *Open Bugs History* (Gráfico 11), podemos observar que, após a *Sprint* 29, houve uma queda dos números de *bugs*.

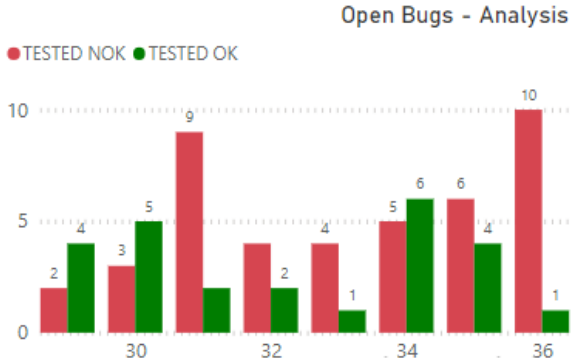
Gráfico 11 – Queda no numero de bugs abertos



Fonte: Elaborado pelo autor

A queda foi ocasionada pela decisão do *Product Owner* de reanalisar os *bugs* do *Backlog* pela equipe de qualidade (a partir da *Sprint* 29). Para isso, foi necessário adicionar um gráfico com o quantitativo de *bugs* analisados por *Sprint* (Gráfico 12) pela equipe de qualidade, visando elucidar o motivo desta queda nas análises das *Sprints* posteriores e, ainda, para demonstrar o quantitativo de *bugs* que foram fechados diante dessa reanálise.

Gráfico 12 – Bugs analisados



Fonte: Elaborado pelo autor

Diante das construções das métricas e gráficos apresentados, houve uma grande satisfação por parte do cliente dos resultados apresentados e tornou-se uma estrutura padrão a ser seguindas e apresentadas na *Sprint Review.* Sendo também apresentado e aplicado para outros projetos do cliente em que possuíam junto a empresa.

6 Conclusões

Após analisar o processo de desenvolvimento e os impactos das métricas criadas ao longo de várias *Sprints*, observou-se que elas foram de fundamental importância para melhorar o processo de desenvolvimento, uma referência para monitoramento e análise e podendo ser aplicada em outro projeto de mesma estrutura na empresa onde o caso foi estudado.

Levando em conta as considerações e análises realizadas no Capítulo 5, torna-se possível observar a importância das métricas para um projeto de desenvolvimento de software.

Diante do acúmulo de papéis do *Scrum* pelo cliente, não recomendado pela literatura e observado no caso estudado, problemas já apontados(Capítulo 3.4), essa participação ativa no processo de desenvolvimento levou a uma diminuição da autonomia de gestão da empresa contratada pelo cliente, o que levou a uma situação de dependência da participação do cliente em todos os eventos do processo, uma contradição evidente com a metodologia ágil. Vale ressaltar que este acúmulo de papéis só foi possível devido à equipe de desenvolvimento ser pequena. Se houvesse um aumento significativo no número de membros, seria necessário rever e desmembrar esse acúmulo de funções pois seria difícil o volume de atividades acumuladas pelo *Product Owner/Scrum Master.*

Na análise referente ao sucesso da *Sprint*, podemos destacar que há a necessidade de cálculo da percentagem do sucesso da *Sprint,* de acordo com o numero de *story points*, e não pelo numero de histórias, visto que as histórias possuem diferentes medidas de esforço. Como efeito da abordagem baseada no número de histórias concluídas, ocorrem divergências entre a produtividade real e aquela registrada no resultado da percentagem, não revelando a eficiência do time corretamente.

Na análise referente ao histórico das *Sprints*, podemos destacar o uso de métricas e parâmetros para poder analisar diferentes cenários pelos quais um projeto pode passar, além de mostrar de forma fácil e sucinta os resultados durante as *Sprints*.

Na análise referente aos *Bugs,* podemos analisar de forma breve e prática as falhas registradas e presentes no *Backlog.* Assim, as métricas adotadas se tornam uma ferramenta útil para a tomada de decisão quanto a solução de *bugs*.

Com a utilização das métricas apresentadas e seus aprimoramentos, é possível ter uma visão geral da evolução do projeto e realizar uma análise clara e rápida por meio da utilização

de recursos gráficos e visuais. Assim, espera-se que estas métricas sejam uma ferramenta para auxiliar as tomadas de decisões, aprimorar processo de desenvolvimento de software em projeto e poder replicar este modelo em outros projetos.

Diante do *feedback* positivo do cliente e consquentemente o constante aprimoramento das métricas, para futuros trabalhos se faz necessário a análise e aplicação desse modelo em outros projetos de diferentes conFigurações e desenvolver um modelo genérico para ser aplicado em uma quantidade maior de projetos e de diferentes áreas.

REFERÊNCIAS

Schwaber, K., & Sutherland, J. (2020). **The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game**. Scrum.org. Acesso em: 3 de julho de 2023.

Cohn, M. (2006). **Agile Estimating and Planning**. Pearson Education.

Kerzner, H. (2017). **Project management: a systems approach to planning, scheduling, and controlling**. John Wiley & Sons.

Turner, J. R., & Müller, R. (2014). **The project manager's leadership style as a success factor on projects: A literature review**. Project Management Journal, 45(6), 21-32.

Pressman, R. S. (2014). **Software engineering: A practitioner's approach**. McGraw-Hill Education.

Donaire Albino, Raphael(2017). **Métricas Ageis: Obtenha melhores resultados em sua equipe**. E-book Casa do código .

Moss, B., & Knight, K. (2019). **Applied Microsoft Power BI: Bring your data to life**! Wiley.

Rad Reza, R. (2019). **Power BI Essentials: An introduction to Microsoft Power BI**. Apress

Guzik, M. (2019). **JIRA Software: A guide for software developers and project managers.** Apress.

WE AGILE YOU. **Planning Poker**. Barcelona, 2022. Disponível em: https://planningpokeronline.com/. Acesso em: 07 de julho de 2023.

ALLIANCE, A. **Manifesto for Agile Software Development**. Utah, 2001. Disponível em:

http://agilemanifesto.org/. Acesso em: 3 de julho de 2023.

ATLASSIAN. **Jira Software.** Sydney, 2022. Disponível em: https://www.atlassian.com/br/software/Jira. Acesso em: 7 de julho de 2023.