UNIVERSIDADE FEEVALE

THIAGO WEBER

DEFINIÇÃO DE PRONTO PARA TIMES *SCRUM*: UM ESTUDO DE CASO BASEADO EM DADOS EM UMA EMPRESA ERP

Novo Hamburgo

2020

THIAGO WEBER

DEFINIÇÃO DE PRONTO PARA TIMES *SCRUM*: UM ESTUDO DE CASO BASEADO EM DADOS EM UMA EMPRESA ERP

Trabalho de conclusão de curso apresentado como requisito parcial à obtenção do grau de Bacharel em Sistemas de Informação pela Universidade Feevale

Orientadora: Prof. Dra. Adriana Neves dos Reis

Novo Hamburgo

2020

AGRADECIMENTOS

Gostaria de agradecer a todos os que, de alguma maneira, contribuíram para a realização desse trabalho de conclusão, em especial:

À Shaiane, por estar presente em todos os momentos com apoio e incentivo a minha dedicação.

À professora Dr. Adriana, pela paciência, pelos ensinamentos e por toda a orientação concedida durante a construção deste trabalho.

Ao professor Dr. Juliano Varella de Carvalho, que se dispôs a ter o perfil de especialista durante a realização do estudo de caso.

À empresa e aos Times *Scrum* participantes do estudo de caso, por oportunizarem o aprendizado.

A todos que, de alguma forma, contribuíram para a conclusão deste trabalho.

Muito obrigado!

RESUMO

Nos dias atuais, as organizações têm o desejo de se tornarem cada vez mais ágeis, sendo o *Scrum* um dos *frameworks* mais utilizados que possui uma forma de desenvolvimento centrada no valor ao cliente e com entregas frequentes. Para isso, ao final de cada *Sprint*, é entregue um incremento de funcionalidade do produto que deve estar na condição de ser utilizado e atender a Definição de Pronto do Time *Scrum*. Devido às dificuldades enfrentadas pelas organizações para fazer a transição de um modelo tradicional para os métodos ágeis, muitos times acabam não possuindo uma Definição de Pronto em que o incremento esteja propenso ao uso e realizam trabalhos adicionais após o término da *Sprint*, o que pode resultar em problemas como falhas descobertas tardiamente, além de fazer com que não haja entregas contínuas e de valor para os clientes. Diante deste contexto, o presente trabalho tem como objetivo, a partir de um estudo de caso realizado em uma empresa ERP que adota o ambiente ágil de desenvolvimento, avaliar o tempo de desenvolvimento e entrega de incrementos do produto para Times *Scrum*. A construção desta proposta baseia-se na revisão da literatura sobre Definição de Pronto em Times *Scrum*, a partir da qual relaciona possíveis restrições organizacionais existentes, identificadas na análise entre o processo atual de negócio e desenvolvimento em que os Times *Scrum* estão inseridos. Essa análise é realizada a partir da Mineração dos Dados de entrega dos incrementos do produto dos Times *Scrum*, gerando previsões e *insights.* A partir da experimentação desta proposta, concluiu-se que a entrega de incrementos do produto dentro das *Sprints* está mais ligada às pessoas e às características dos times analisados, juntamente com o contexto no qual estão empregados.

Palavras-chave: Definição de Pronto; Mineração de dados; *Scrum;* ERP.

ABSTRACT

Nowadays, the organizations have the desire to become more and more agile, Scrum being one of the most used frameworks that has a form of development centered on customer value and frequently. For this, at the end of each Sprint, an increase in the functionality of the product is provided, which must be subject to the condition of use and compliance with the Definition of Done of the Scrum Team. Accepting the difficulties faced by organizations in making the transition from a traditional model to methods, often ended without having a Definition of Done in which or increment ready to use and achievements after the end of Sprint, or that can be used in problems such as flaws discovered late, in addition to ensuring that there are no continuous and valuable deliveries for customers. Given this context, the present work aims, based on a case study carried out in an ERP company that adopts the agile development environment, to evaluate the development time and delivery of product increments to Scrum Times. The construction of this proposal is based on the literature review on Definition of Done in the Scrum Times, from which alternative relationship is authorized to allow changes identified in the analysis between the current business and development process in which the Scrum Times is inserted. This analysis is performed from the Mining of delivery data of the product increments of the Times Scrum, generating recovery and insights. From the experimentation of this proposal, it was concluded that the delivery of product increments within Sprints is more linked to the people and the characteristics of the analyzed times, associated with the context in which they are employed.

Keywords: Definition of Done; Data Mining; Scrum; ERP.

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BI	*Business Intelligence*
CRM	*Customer Relationship Management*
DoD	*Definition of Done*
ERP	*Enterprise Resource Planning*
FDD	*Feature Driven Development*
ID	*Identity*
KDD	*Knowledge Discovery in Databases*
PB	*Product Backlog*
PO	*Product Owner*
SM	*Scrum Master*
XP	*eXtreme Programming*

SUMÁRIO

1 INTRODUÇÃO

Hoje em dia as organizações têm o desejo de se tornarem mais ágeis para resultar em uma maior eficiência, conforme indicam Prikladnicki, Willi e Milani (2014), e o *Scrum* é o *framework* que se destaca, sendo o mais utilizado entre os métodos ágeis conhecidos no mercado, segundo pesquisa do Version One (2019). De acordo com Rubin (2012), ele tem uma forma de desenvolvimento centrada no valor ao cliente, e foi criado para auxiliar no desenvolvimento de produtos complexos dentro de ambientes complexos (SABBAGH, 2014).

As partes entregues do produto são as mais necessárias para os clientes e usuários no momento da entrega, sendo geradas em ciclos curtos de desenvolvimento que ocorrem de forma sequencial que fazem com que sejam utilizadas imediatamente (SABBAGH, 2014). Como resultado, os clientes obtêm um fluxo contínuo de recursos com maior valor de forma mais rápida, o qual é focado na entrega de recursos úteis, integrados, testados e valiosos (RUBIN, 2012).

As entregas devem acontecer com frequência, de acordo com os subconjuntos avaliados pelo cliente de todo o produto. Ao final de cada *Sprint*, o time de desenvolvimento entrega um incremento de funcionalidade do produto que deve estar na condição de ser utilizado e atender a Definição de Pronto, em inglês *Definition of Done* (DoD), do Time *Scrum* (SCHWABER; SUTHERLAND, 2017). Sempre que esse incremento, ou a soma desses, representa valor suficiente e já pode ser utilizado, é importante que chegue a seus usuários o mais rápido possível. Por meio de *releases* frequentes, o Time *Scrum* pode obter *feedback* dos usuários do produto e, assim, reduzir os riscos e produzir o produto certo, além de conseguir dar um senso de progresso do projeto aos seus clientes e demais partes interessadas, provendo retorno ao investimento realizado por eles (SABBAGH, 2014).

Neste contexto, a aplicação do *Scrum* pode depender do projeto, da organização, dos *stakeholders* e da experiência do time, contribuindo para a natureza complexa da adoção do *framework* (CUNHA; ANDRADE, 2014). A transição de um modelo tradicional para os métodos ágeis é difícil, exigindo diversas mudanças para se obter os benefícios que o ágil pode trazer. Também há muita demanda, não apenas dos desenvolvedores, mas do restante da empresa (COHN, 2011).

Dentre essas dificuldades está o fato de o Time *Scrum* determinar a sua Definição de Pronto, sendo este um acordo entre o *Product Owner* (PO) e o time de

desenvolvimento sobre o que deve ser atendido para que um item, ou o incremento do produto como um todo, seja considerado pronto.

A definição é a mesma para todos os itens do *Product Backlog* (PB) e estabelece que o resultado do trabalho de um time de desenvolvimento em uma *Sprint* seja entregável. Assim, o ideal é que nenhum desenvolvimento, revisão de código, teste, integração ou quaisquer tarefas adicionais devam ser necessárias após o time considerar o incremento do produto produzido na *Sprint* pronto, para que o mesmo possa ser entregue aos clientes finais.

Para a construção de uma Definição de Pronto efetiva é importante que se considerem as restrições organizacionais que afetam o trabalho do Time *Scrum*, sejam restrições de negócio, de processo, tecnológicas ou culturais (SABBAGH, 2014). Devido a essas restrições, em muitos Times *Scrum*, é após o término da *Sprint* que o resultado do trabalho se torna entregável, através do desenvolvimento de algum trabalho adicional (SABBAGH, 2014). A existência desse trabalho adicional é uma disfunção e traz riscos consideráveis ao projeto, sendo que diversos problemas serão descobertos tardiamente, ou seja, o fato de a Definição de Pronto não ser bem estipulada no *Scrum* pode resultar em um problema, pois não haverá as entregas contínuas e de valor para os clientes.

Segundo Sabbagh (2014), na medida do possível, o Time *Scrum* trabalha ao longo do projeto para reduzir essa disfunção, transferindo progressivamente esse trabalho para dentro das *Sprints* e, assim, tornando sua Definição de Pronto cada vez mais estrita. Como resultado, podem começar com um estado final menor, em que os recursos não estão totalmente concluídos e deixar sua Definição de Pronto evoluir com o tempo, à medida que impedimentos organizacionais são removidos (RUBIN, 2012).

Entretanto, nem sempre os times conseguem levar à prática o que está descrito na teoria, e quando a equipe se depara com uma prática de difícil aplicação em seu contexto, muitas vezes, prefere alterar o *Scrum* ao invés de se adaptar ao apresentado pelo método, tornando-se um obstáculo na aplicação da ferramenta (PRIKLADNICKI; WILLI; MILANI, 2014). Segundo Cohn (2011), mudar práticas é um caso, mas mudar o modo de pensar é outro bem diferente. Portanto para a adoção de métodos ágeis obter sucesso é preciso que toda cultura organizacional se adapte à teoria.

Outro fator importante são as coletas de métricas, que: trazem vantagens como o combate ao retrocesso gerado com o tempo pela inércia organizacional, aumentam

o esforço dos colaboradores ao visualizarem os sucessos iniciais que vão sendo obtidos durante transição e ajudam a saber para onde direcionar tentativas adicionais de melhorias. Apesar dos benefícios, na indústria de *softwares* não se tem um longo histórico de se realizar tais medidas porque é trabalhoso, mesmo as mais simples de serem obtidas (COHN, 2011).

Partindo desta problemática, o presente trabalho visa avaliar o tempo de desenvolvimento e entrega de incrementos do produto para Times *Scrum* que desenvolvem *Enterprise Resource Planning* (ERP). Ele é baseado na análise de dados que demonstrem o ciclo completo do produto, iniciando na solicitação do cliente e sendo concluído em sua entrega, gerando *insights* para o problema em questão.

Para atender a proposta são consideradas as peculiaridades de se desenvolver um sistema ERP com o *Scrum*, sendo o mesmo um tipo completo de sistema para apoio à decisão gerencial, contemplando os mais diferentes módulos para esse fim. Também vale ressaltar que esse tipo de *software* costuma sofrer modificações diárias e em grande volume, muitas vezes caracterizadas por correções de falhas e alterações legais para adequação à realidade fiscal (SOUZA E SILVA, 2015).

Como Cohn (2011) resumiu, ágil não é algo que se torna, mas é algo que se torna mais. Ao tomar a decisão de migrar para essa metodologia, é importante estar ciente de que não se chegará ao estado final de forma rápida, e alguns processos, como a entrega do produto pelo time desenvolvimento, podem precisar ser revistos. Tornar-se mais proficiente com o *Scrum* e mais ágil é um processo de melhoria contínua, visando sempre a melhora em seus resultados (RUBIN, 2012).

1.1 OBJETIVOS

1.1.1 Objetivo geral

O objetivo deste trabalho é realizar uma análise preditiva do tempo de desenvolvimento e entrega de incrementos do produto a partir da Mineração de Dados, gerando previsões e insights baseados em dados coletados, através de um estudo de caso em Times *Scrum* que desenvolvem um sistema ERP.

1.1.2 Objetivos específicos

- Realizar uma revisão na literatura sobre Definição de Pronto em Times *Scrum*;
- Mapear o processo atual de negócio e desenvolvimento dos times, relacionando restrições organizacionais existentes com a metodologia *Scrum*;
- Realizar a Mineração dos Dados de entrega dos incrementos do produto dos Times *Scrum* para fazer uma análise preditiva e gerar previsões e *insights*;
- Efetuar uma validação nos resultados identificados na análise;
- Aplicar um questionário nos times envolvidos, buscando identificar a percepção com suas Definições de Pronto.

1.2 METODOLOGIA

A pesquisa, quanto à sua natureza, se enquadra como pesquisa aplicada, pois visa gerar conhecimentos que serão aplicados a um problema específico. Quanto aos objetivos, o estudo possui caráter exploratório, já que ele busca investigar e prover mais informações sobre o assunto da pesquisa durante o estudo (PRODANOV; FREITAS, 2013).

Em relação aos procedimentos para a pesquisa, são utilizadas: revisão da literatura e estudo de caso. O trabalho tem como início a realização de uma revisão da literatura sobre Definição de Pronto em Times *Scrum*. Além disso, foi feita uma busca por trabalhos correlatos em que tenham sido realizados estudos sobre prontos definidos por Times *Scrum* para o produto em outros cenários, para identificar semelhanças e diferenças em relação ao ERP. Já o estudo de caso é utilizado na análise entre o processo atual de negócio e desenvolvimento em que Times *Scrum* estão inseridos, avaliando múltiplos casos de times e definições de pronto, relacionando possíveis restrições organizacionais existentes com a metodologia *Scrum*.

O planejamento dos casos é focado em uma empresa ERP que adotou o ambiente ágil de desenvolvimento. Inicia com a realização de uma coleta dos dados que refletem no tempo de desenvolvimento e entrega dos incrementos do produto, gerando um *dataset* que foi preparado para posterior análise. A realização dessa

preparação dos dados seleciona os atributos mais relevantes, com dados quantitativos e relacionados às etapas e pessoas envolvidas no processo, limpa os dados inconsistentes e transforma os tipos de dados.

A aplicação de técnica de Mineração de Dados gera matrizes de confusão e árvores de decisão. Através da execução de algoritmos desenvolvidos se realiza o processamento dos dados, efetuando análises preditivas para geração de previsões e *insights*.

O estudo tem uma abordagem que pode ser considerada como qualitativa, uma vez que o ambiente natural é a fonte direta para a coleta e interpretação dos dados (PRODANOV; FREITAS, 2013).

Por fim é realizada a avaliação e interpretação dos resultados de todo o processo, conferindo se foram encontrados os mesmos comportamentos identificados na análise. Também é aplicado um questionário com os times envolvidos no estudo, buscando identificar a percepção do time e seu nível de satisfação com suas definições de pronto.

1.3 ESTRUTURA DO TRABALHO

Primeiramente, no Capítulo 2, são destacados conceitos sobre sistema ERP, incremento do produto e Definição de Pronto, os quais caracterizam-se como as evidências coletadas na literatura. Em seguida, no Capítulo 3, é apresentada a proposta e, a partir da mesma, é descrito o estudo de caso, juntamente com o processo da empresa. No Capítulo 4 são apresentadas as avaliações realizadas sobre os resultados obtidos e aplicação dos questionários. Por fim, as conclusões sobre a pesquisa são expostas.

2 REFERENCIAL TEÓRICO

Este capítulo tem por objetivo trazer os conceitos e características de um sistema ERP, assim como, revisar alguns conceitos relacionados a entregas em ambiente ágil que precisam ser inseridos nesse ambiente. A partir desses referenciais são apresentados trabalhos relacionados ao tema proposto, mostrando diferentes cenários de Definição de Pronto entre organizações que adotaram *Scrum.*

2.1 ERP

Para iniciar o estudo proposto é preciso definir o que é um sistema ERP, identificar suas peculiaridades e as características das organizações desenvolvedoras desse tipo de *software*.

A sigla ERP traduzida literalmente significa Planejamento dos Recursos da Empresa, o que não explica exatamente o que esse sistema faz. No Brasil ele é chamado de Sistemas Integrados de Gestão Empresarial, que condiz melhor com seu significado. Como explica Souza e Silva (2015), ele tem como principal característica a integração de todas as informações e módulos da empresa e é considerado uma das mais completas ferramentas de apoio à decisão gerencial.

Um ERP é um sistema de gestão integrado que processa as informações da organização em um único banco de dados. Possui um fluxo de dados em tempo real que faz com que sejam eliminados muitos processos redundantes, sendo considerado um instrumento para a melhoria do processo de negócios. O sistema ERP é capaz de visualizar por completo as transações da empresa, desenhando um amplo cenário dos negócios (CHOPRA; MEINDL, 2003).

Por essas características ele é muito valorizado pelas organizações, como indica Chopra e Meindl (2003). Devido à significativa concorrência, o ambiente empresarial atual estipula metas de constante crescimento e melhoria contínua, tornando-se necessária uma ferramenta que faça a integração de todas as informações para se obter informações consolidadas com maior precisão (SOUZA E SILVA, 2015).

Algumas das características de um ERP, segundo Souza e Silva (2015), são:

- Banco de dados unificado: possui integração das informações, fazendo com que a velocidade de processamento e resposta sejam rápidas e as informações sejam processadas uma única vez;

- Personalização: possui capacidade de se adaptar às necessidades relativas a cada empresa, tornando-se mais flexível e ajustável aos processos das empresas;
- Arquitetura simples: maneira como as informações são dispostas em sua arquitetura de *software*, facilitando o fluxo da informação e novas implementações.

Ao adotar um ERP, o objetivo básico não é colocar o software em produção, mas melhorar os processos de negócios usando tecnologia da informação, implicando um processo de mudança organizacional. As vantagens são importantes tanto para as empresas quanto para as pessoas envolvidas no processo, sejam ela, diretores, funcionários ou clientes. Algumas dessas vantagens são: suporte à tomada de decisão; valor agregado ao produto; produtos de melhor qualidade; oportunidade de negócios e aumento da rentabilidade; mais segurança nas informações, menos erros, mais efetividade e produtividade; carga de trabalho reduzida; redução de custos e desperdícios (SILVA; OLIVEIRA, 2016).

Já para a empresa desenvolvedora, é comum, segundo Araújo e Dielle (2015), que o *software* sofra modificações diárias e em grande volume, muitas vezes caracterizadas por correções de falhas oriundas de modificações recentes. Além disso, há frequentemente uma pressão extra que são as alterações legais para adequação do *software* à realidade fiscal, que está sempre a mudar.

2.2 INCREMENTO DO PRODUTO

O incremento do produto consiste na soma de todos os itens da *Sprint Backlog* que foram concluídos na *Sprint* (SABBAGH, 2014). É um incremento de funcionalidades do produto prontas, composto por novas funcionalidades e por melhorias no que foi produzido anteriormente (SABBAGH, 2014).

Segundo Cohn (2011), espera-se que os Times *Scrum* entreguem algo de valor para os usuários ou clientes a cada *Sprint*, possibilitando que seja obtido *feedback* sobre o que foi feito na última iteração. O incremento do produto é demonstrado na *Sprint Review* pelo time de desenvolvimento e o PO (SABBAGH, 2014).

Nem sempre um incremento representa valor suficiente para ser utilizado por seus usuários e o PO pode optar por acumular alguns incrementos para só então

realizar a entrega aos interessados. Ainda que represente valor suficiente, diferentes razões podem justificar que não seja feita a entrega ao final de cada *Sprint* como questões políticas, burocráticas ou técnicas, ou então quando se acredita que os clientes podem não conseguir absorver mudanças tão frequentemente (SABBAGH, 2014). O PO é quem deve decidir se será liberado ou não o incremento por ser o dono do produto, tendo o papel responsável pelo gerenciamento do mesmo (SCHWABER; SUTHERLAND, 2017). A seguir serão apresentadas algumas características que um incremento do produto com relação a sua entrega.

2.2.1 INCREMENTO DE PRODUTO PASSÍVEL DE SER ENTREGUE

Todo Time *Scrum* deve discutir e entrar em acordo sobre uma Definição de Pronto que estabeleça um incremento de produto passível de ser entregue apropriado ao seu ambiente. Então cada item trazido do PB para uma *Sprint* deve satisfazer os critérios definidos antes de ser considerado terminado (COHN, 2011).

Embora caiba à empresa ou à equipe estabelecer uma definição apropriada de concluído para seu contexto, segundo Cohn (2011), há certas diretrizes que são aplicáveis à maioria dos projetos *Scrum* em quase todas as empresas:

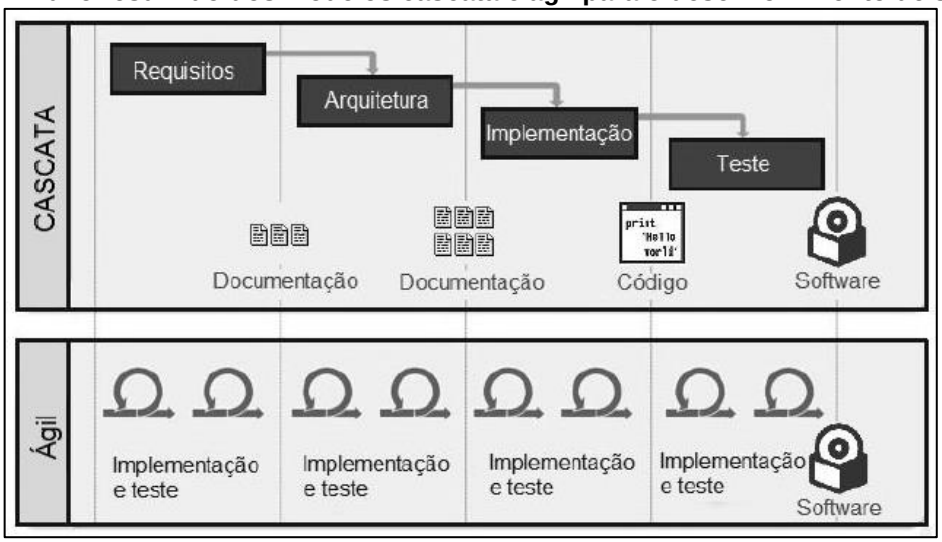
- Passível de ser entregue significa testado: No final da *Sprint* deve-se ter a expectativa de que os novos requisitos não tenham erros e de que nenhum erro tenha sido introduzido em requisitos antigos, então, não há como imaginar uma situação em que seja correto a equipe deixar o teste fora dessa definição;
- Passível de ser entregue não significa necessariamente coeso: Só porque um produto é passível de ser entregue não significa que alguém já esteja querendo que ele seja entregue. Às vezes pode levar duas, três ou mais *Sprints* para que um conjunto de requisitos esteja integrado de uma maneira minimamente útil;
- Passível de ser entregue significa integrado: Em um projeto de várias equipes, elas devem definir como concluído de tal forma que seja inclusa a integração de fluxos de desenvolvimento.

2.2.2 ENTREGA DE PRODUTOS

Em projetos que utilizam métodos cascata, apenas uma ou poucas entregas são realizadas ao final do projeto ou ao final de uma grande etapa, conforme mostra

a Figura 1. Segundo Sabbagh (2014), as partes entregues são as mais necessárias para seus clientes e usuários no momento da entrega e, por essa razão, serão utilizadas imediatamente. Uma vez que são utilizadas, representam retorno ao investimento realizado. Da mesma forma, *Scrum* possibilita um curtíssimo prazo para a introdução do produto no mercado, já que cada uma dessas entregas, somada às anteriores, representa um produto funcional.

Figura 1 - Fluxo resumido dos modelos cascata e ágil para o desenvolvimento de *software*



Fonte: Gomes (2014)

Gomes (2014) explica que, no desenvolvimento ágil de *software*, o time inteiro está envolvido em definir requisitos, otimizá-los, implementá-los, testá-los, integrá-los, e entregá-los aos clientes. Outra característica do ágil é o fato de que planejar é uma atividade constante, sendo possível responder rapidamente às mudanças e realizar as adaptações que forem necessárias para que o *software* seja entregue com qualidade e eficiência.

Todo o ciclo de execução de uma *Sprint* foca em agregar valor ao produto, um *potentially shippable product increment*. Mas, às vezes, alguma entrega depende de autorização do negócio que envolve questões publicitárias, comerciais, parceiros ou alguma estratégia de lançamento. É importante que o time esteja a par de forma a estimar sua participação direta ou indireta quanto à entrega (AUDY, 2015).

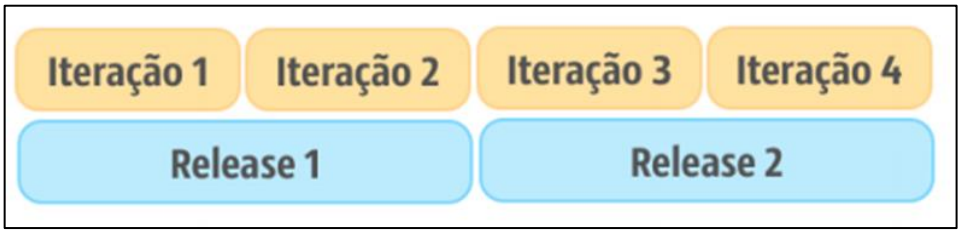
2.2.3 PLANEJAMENTO DE *RELEASES*

Um dos principais benefícios dos métodos ágeis é a redução do tempo das entregas de iterações para o cliente. Segundo Gomes (2014), durante o planejamento

de *releases* é importante descobrir qual o escopo necessário para agregar algum valor ao cliente e, então, definir uma *release* para entrega, procurando potencializar o *time-to-market*.

Nas *releases*, o *software* é entregue para o cliente pronto para produção. Como mostra a Figura 2, as iterações representam um determinado intervalo de tempo em que algumas histórias serão implementadas em ciclos que estão contidos dentro das *releases* (GOMES, 2014).

Figura 2 - Estrutura de *releases* do *software*



Fonte: Sabbagh (2014)

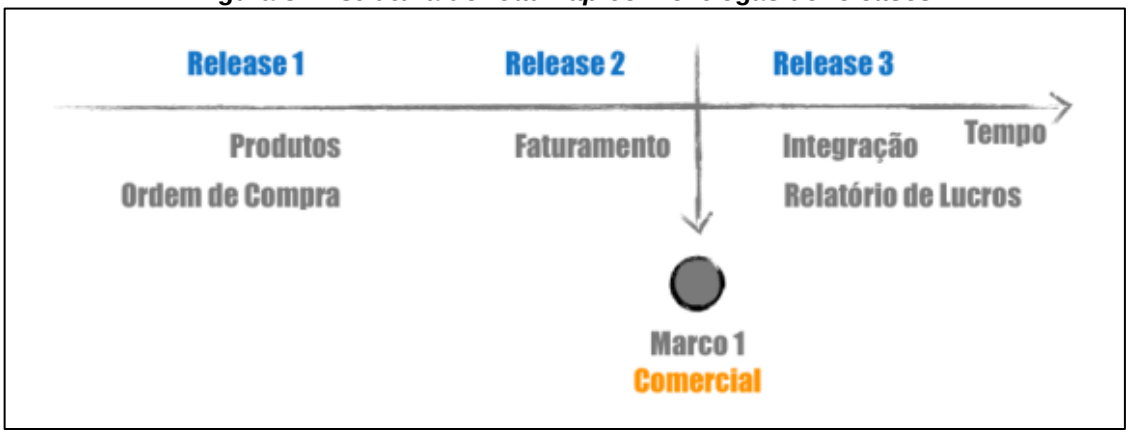
Conforme explica Sabbagh (2014), *release* é um objetivo ou uma necessidade de negócios de alto nível a ser alcançada pelo time de desenvolvimento para uma entrega, representando um passo em direção à visão do produto. Quanto menores forem os ciclos de *release*, mais rápido o cliente poderá se beneficiar do produto, e oferecer *feedback* para que ele possa ser melhorado (GOMES, 2014).

Para definir quando novos incrementos do projeto serão instalados em produção, pode-se realizar reuniões de planejamento de *releases* a cada duas semanas ou mensalmente. Todos os interessados do projeto podem participar da reunião, mas é importante que ela seja composta por todos que estiverem relacionados ao processo de entrega, como especialistas de integração, especialistas de redes, desenvolvedores, testadores e *designers*, além das pessoas capazes de tomar decisões técnicas, de negócio e gerenciais (GOMES, 2014).

2.2.4 *ROADMAP* DO PRODUTO

O *Roadmap* é uma ferramenta útil para traçar um plano de alto nível sobre *releases* e marcos importantes do produto que serão alcançados ao longo do tempo. Como exemplifica a Figura 3, oferece visibilidade de que funcionalidades serão entregues primeiro e quais serão entregues depois e em quais *releases* (GOMES, 2014).

Figura 3 - Estrutura do *roadmap* com entregas de *releases*



Fonte: Sabbagh (2014)

Como o desenvolvimento de *software* é complexo e difícil de se estimar, num contexto ágil o *roadmap* é uma ferramenta útil para mostrar intenções de *release* e apresentar marcos do produto. Porém não possui a mesma utilidade para definir prazos exatos de entrega de funcionalidades, justamente porque é natural que mudanças no escopo do produto ocorram ao longo do desenvolvimento, conforme explica Gomes (2014). Outra utilização, segundo Sabbagh (2014), ocorre quando se trabalha entrega contínua e não faz sentido se ter *releases* maiores, sendo que elas são entregues em cada *Sprint* e correspondem ao valor de negócio do item entregue.

2.3 CAUSAS DE ATRASOS EM PROJETOS

Segundo Cepeda, Coutinho e Vigna (2017), as causas de atrasos em projetos de *software* mais relevantes são:

- Estimativas imprecisas de prazo/custo/qualidade;
- Excesso de burocracia na organização;
- Mudanças no escopo sem alteração do *baseline*;
- Inadequado planejamento dos recursos;
- Cronograma definido sob pressão do cliente.

Dentre essas causas, a aplicação de metodologias ágeis pode aportar uma estrutura de gestão menos burocrática, reduzindo a quantidade de documentação após análise e adequação da empresa, e mais focada em resultados, diminuindo ao longo do tempo as falhas relacionadas ao desenvolvimento, como explicam os mesmos autores. Também pode diminuir os problemas relacionados ao planejamento

e às mudanças no escopo, uma vez que possui ciclos curtos de planejamento e desenvolvimento.

No *Scrum*, um dos principais objetivos é ser ágil, adaptável e rápido. Com base em um modelo prioritário e incremental de entrega, no qual os recursos de maior valor são continuamente construídos e entregues na próxima iteração, os clientes obtêm um fluxo contínuo de recursos de alto valor mais rapidamente. Porém não se pode confundir rapidez com ser apressado, pois violaria um dos princípios que é o ritmo sustentável. As pessoas devem ser capazes de trabalhar em um ritmo em que possam continuar por um longo período evitando que se perca qualidade na entrega (RUBIN, 2012).

2.4 DEFINIÇÃO DE PRONTO

O foco do *Scrum* é que sejam entregues recursos úteis, integrados, testados e valiosos, e que cada iteração leve a resultados entregues rapidamente (RUBIN, 2012). Segundo Sutherland (2014), para cada história que se deseja fazer deve haver uma Definição de Pronto que deve ser independente, negociável, valiosa, estimável, pequena, testável.

A Definição de Pronto é um acordo formal entre PO e Time de Desenvolvimento sobre o que é necessário para que um item ou o Incremento do Produto como um todo seja considerado pronto e, assim, passe a fazer parte do produto em desenvolvimento (SABBAGH, 2014). São critérios definidos por ambos para garantir que se tenha um entendimento compartilhado do que significa o trabalho estar completo, assegurando a transparência, e para que nada de importante seja esquecido ou deixado de lado (SCHWABER; SUTHERLAND, 2017). A participação do PO na construção dessa definição é muito importante, uma vez que ele conhece melhor do que ninguém as expectativas do cliente (GOMES, 2014).

A mesma Definição de Pronto se aplica para todos os itens do PB que representem funcionalidades a serem desenvolvidas e guia o Time de Desenvolvimento durante o trabalho na *Sprint*, sendo utilizada para garantir que o Incremento do Produto gerado na *Sprint* seja entregável. Apenas os itens prontos de acordo com a Definição de Pronto podem ser apresentados na reunião de *Sprint Review* (SABBAGH, 2014).

A Definição de Pronto é criada antes do início do desenvolvimento do produto, podendo ser modificada e evoluída ao longo do desenvolvimento do produto, à medida

que novas necessidades sejam identificadas ou que se aproxime do “pronto” e “entregável” (SABBAGH, 2014). Ela não deve ser estática e é natural que, à medida que o time for evoluindo e amadurecendo, seja alterada para atender critérios de maior qualidade (GOMES, 2014).

Porém, segundo Rubin (2012), muitas equipes começam com uma definição que não termina em um estado onde todos os recursos estão concluídos, por existirem impedimentos reais que podem dificultar a chegada a esse estado. Com isso, elas podem começar com um estado final menor e deixam sua Definição de Pronto evoluir conforme os impedimentos são removidos. Conforme Schwaber e Sutherland (2017), em um Time *Scrum* maduro é esperado que a sua Definição de Pronto seja expandida para incluir critérios mais rigorosos de alta qualidade.

A Definição de Pronto geralmente se parece com uma lista de atividades do que deve ser feito antes que uma história possa ser considerada potencialmente entregável (SABBAGH, 2014). Segundo Gomes (2014), essas atividades incluem tudo o que deve ser realizado para se construir o produto, onde cada equipe deverá examinar o seu contexto, ou seja, levando em consideração a natureza do produto, a tecnologia que está sendo utilizada, as necessidades dos usuários etc. Alguns exemplos são:

- Código fatorado;
- Código dentro dos padrões de codificação;
- Código revisado ou feito em par;
- Código integrado no sistema de controle de versão;
- Documentação de arquitetura atualizada;
- Testes de unidade realizados;
- Testes de aceitação realizados;
- Testes exploratórios realizados;
- Nenhum defeito conhecido pendente;
- Manual do usuário atualizado.

A Definição de Pronto varia de time para time, de projeto para projeto. O time de desenvolvimento possui, entre seus integrantes, todas as habilidades e conhecimentos necessários para entregar, ao final da *Sprint*, um incremento do produto pronto de acordo com a Definição de Pronto (SABBAGH, 2014).

Para a construção de uma Definição de Pronto é importante levar em consideração possíveis restrições organizacionais que afetam o trabalho do Time *Scrum*. Essas podem ser de negócio, de processo, tecnológicas ou culturais, e devido a elas muitos times iniciam o projeto com uma Definição de Pronto distante da ideal: após o término da *Sprint*, algum trabalho adicional ainda será necessário para tornar o resultado do trabalho entregável. A existência desse trabalho adicional não é ideal, pois traz riscos consideráveis ao projeto, já que diversos problemas podem ser descobertos tardiamente (SABBAGH, 2014).

Conforme Rubin (2012) esse trabalho adicional deve ser feito em algum momento, e por vezes pode demorar mais do que a duração de uma *Sprint*. Como explica Sabbagh (2014), na medida do possível o Time *Scrum* trabalha para transferir progressivamente trabalhos adicionais para dentro das *Sprints* e assim tornar sua Definição de Pronto cada vez mais estrita. Porém existem casos, por exemplo projetos de alto risco, em que são necessários testes adicionais e auditorias externas, onde o trabalho adicional após conclusão da *Sprint* existirá durante todo o projeto.

O time de desenvolvimento entrega um incremento de funcionalidade do produto a cada *Sprint* e este incremento é utilizável, então o PO pode escolher por liberá-lo imediatamente. Se a Definição de Pronto para um incremento é parte das convenções, padrões ou diretrizes de desenvolvimento da organização, todos os Times *Scrum* devem segui-la como um mínimo (SCHWABER; SUTHERLAND, 2017).

Conforme os princípios do Manifesto Ágil, a prioridade é satisfazer o cliente, através da entrega adiantada e contínua de *software* de valor, e entregar *software* funcionando com frequência, na escala de semanas até meses, com preferência aos períodos mais curtos (BECK et al., 2001). Porém, como relata Rubin (2012), pronto não significa que o que foi construído deve ser enviado. A entrega é uma decisão de negócio que frequentemente ocorre em uma cadência diferente; sendo que em algumas organizações pode não fazer sentido no final de cada *Sprint*.

Segundo Rubin (2012), algumas equipes adotam o conceito *done-done*, que é um termo utilizado para indicar que o trabalho realizado durante a *Sprint* foi realmente realizado:

- *Done*: fazer tanto trabalho quanto a equipe está preparada para fazer;
- *Done-done*: fazer o trabalho necessário para ser entregue aos clientes.

As equipes que usam o termo *done* costumam utilizá-lo para indicar que foi realizado tanto trabalho quanto o time estava preparado para fazer. Se aplica em equipes que possuem restrições e não conseguem concluir a *Sprint* com o incremento pronto para ser entregue. Já equipes ágeis maduras não possuem esta distinção de conceitos de pronto, pois conseguem realizar a entrega do incremento do produto dentro da *Sprint,* ou seja, estão aptas a realizar todas as tarefas necessárias para concluir o trabalho (RUBIN, 2012).

2.5 TRABALHOS RELACIONADOS

Recentemente foram realizados alguns estudos de casos em empresas desenvolvedoras de *software* ERP que adotaram o *Scrum* que mostram diferentes cenários de Definição de Pronto entre as organizações. Entre eles destacam-se as peculiaridades de cada empresa, o que implica distintos problemas para implantação do *Scrum*.

Araújo e Dielle (2015) focaram seu estudo desde o início da adoção do *Scrum* em uma empresa ERP, avaliando a aceitação da mudança e todos os problemas encontrados. A motivação da empresa pela adoção do *Scrum* foi decorrente de um esforço iniciado pela gerência e por pessoas em cargos de liderança, para retomar o controle das modificações que estavam sendo realizadas no produto. Esse *software* sofria modificações diárias e, em resumo, se mantinha basicamente para corrigir suas próprias falhas. Devido ao tempo gasto com retrabalho, os clientes estavam recebendo menos alterações do que demandavam.

A empresa possui sua própria classificação de solicitações de alterações, sendo elas:

- manutenções corretivas: erro no sistema;
- manutenções adaptativas: alteração legal;
- manutenções evolutivas: sugestão de melhorias, complemento de tarefa, orçamento e solicitação de novo recurso.

Outras classificações de alterações realizadas pela empresa não foram atribuídas a nenhum tipo, pois não estão associadas a mudanças no sistema ERP, como: liberação de módulos, documentação do *help* e treinamento interno.

Com a implantação do *Scrum* algumas mudanças foram percebidas. Com relação à entrega de demandas, ficou definido chamar de concluídas as alterações

que foram entregues ao final da *Sprint* e não necessariamente aprovadas pelo controle de qualidade, que é o setor da empresa responsável pelos testes internos do produto. Caso a alteração gerasse uma falha que fosse oriunda dos testes internos ou de clientes, sua correção era solicitada em uma nova solicitação de alteração, nem sempre corretamente associada àquela que a originou.

Mesmo se todas as solicitações forem consideradas como concluídas ao término da *Sprint*, o controle de qualidade poderá ainda, nos dias seguintes (ou seja, durante o seguimento das *Sprints* seguintes), reprovar aquelas nas quais encontrar defeito ou inconformidade, podendo elas serem originadas de qualquer *Sprint* (anteriores ou atual). Ou seja, nesse estudo de caso a entrega das demandas de uma *Sprint* vai sendo feita à medida que mais e mais de suas solicitações vão sendo testadas pelo setor de controle da qualidade.

Outro dado interessante é a relação entre o volume de solicitações corretivas, adaptativas e evolutivas, antes e após a aplicação do *Scrum*, a qual é apresentada no Quadro 1.

Quadro 1 - Comparativo entre os volumes de solicitações após a aplicação do *Scrum*

Tipo de solicitação	Antes do *Scrum*	Após o *Scrum*
Corretivas	77%	74%
Adaptativas	3%	4%
Evolutivas	20%	22%

Fonte: Araújo e Dielle (2015)

Em resumo, após a aplicação do *Scrum,* o *software* desenvolvido não é liberado de forma funcional e contendo valor para os usuários ou clientes após cada *Sprint*, ou seja, não possui uma Definição de Pronto que pode ser entregue. Porém a aplicação de novas técnicas de desenvolvimento e melhoria ou atualização dos sistemas internos de controle, apesar de não terem contribuído para diminuir o volume solicitações de correção de erros, foram importantes para compor o cenário atual da empresa, que possui códigos muito mais fáceis de manter e opera com maior tranquilidade devido à baixa urgência das solicitações.

Já Silva e Oliveira (2016) elaboraram um estudo mais focado no desenvolvimento de um ERP específico para um ou mais clientes, e não em um ERP já consolidado que recebe manutenção constante. Para esse caso, foi aplicado uma mistura de conceitos XP (*eXtreme Programming*), *Scrum* e FDD (*Feature Driven Development*).

Utilizou o artefato do *Scrum* de PB para separar as demandas por módulos do ERP como se fossem demandas de vários times, porém um único time que trabalha no desenvolvimento do *software*. O principal diferencial nesse caso é que possui *Sprints* para desenvolvimento, mas a entrega ocorre somente em períodos de 45 a 60 dias, onde possui implementações e correções de erros. Ou seja, não possui uma Definição de Pronto dentro das *Sprints*, mas ainda assim os *stakeholders* ficam motivados por saberem quando a entrega ocorre e conseguirem visualizar e redefinir o que é necessário no produto que está sendo desenvolvido.

Quadros (2017) realizou um estudo focado na aplicação de *Scrum* e *Kanban* em uma empresa desenvolvedora de ERP. Possui etapas além do desenvolvimento que não dependem do time para conclusão da tarefa e com isso sua Definição de Pronto, que é “em produção”, ou seja, já passou pela validação e está funcionando no cliente, nem sempre é concluída em uma *Sprint*.

Com o *Kanban* é possível realizar a visualização imediata da posição de determinado cartão em meio a um processo. Utilizando o *Kanban* em conjunto ao *Scrum* pode-se visualizar diretamente em que ponto está cada integração e quais serão os gargalos para a entrega de determinada *Sprint*, ou até para a entrega do projeto como um todo.

Pontos definidos pela empresa que são necessários para realização do projeto:

- Planejado: todos os cartões dentro desse quadro ainda não foram iniciados, sendo assim, estão aguardando o término de determinada *Sprint* para iniciar.
- Desenvolvimento: todos os cartões dentro desse quadro estão em desenvolvimento e dentro de uma *Sprint*.
- Homologação: todos os cartões dentro desse quadro estão aguardando a homologação do usuário.
- Aguardando entrada em produção: todos os cartões dentro desse quadro estão à espera de entrar em produção, dessa forma, esses cartões já foram validados, porém não entregues.
- Em produção: neste ponto o cartão está finalizado e foi entregue.

2.6 CONSIDERAÇÕES DO CAPÍTULO

Este capítulo finaliza o conteúdo proposto em seus objetivos específicos quanto ao referencial teórico deste trabalho. Tendo em vista que o desenvolvimento de um *software* ERP possui como peculiaridade o fato de, por vezes, não conseguir planejar com muita antecedência algumas demandas, a aplicação de uma metodologia ágil como o *Scrum* nesse tipo de ambiente pode ter algumas dificuldades. Além disso, enriqueceu os conhecimentos sobre incremento do produto e Definição de Pronto para aplicação do estudo de caso com os conceitos que a literatura oferece ao respectivo assunto de pesquisa.

A partir dos trabalhos relacionados citados, percebe-se que o problema está presente nos mais diferentes contextos de desenvolvimento de *software*. Assim, seguindo o planejamento estrutural deste trabalho, no próximo capítulo será feita uma contextualização do problema e detalhamento do estudo de caso realizado.

3 ESTUDO DE CASO

O conceito de Definição de Pronto, segundo apontado no capítulo 2, indica que a cada *Sprint* deve ser possível entregar um incremento do produto funcional ao usuário final. Porém, nem sempre é possível ou interessante que essa entrega seja feita dentro da própria *Sprint* ou imediatamente após a mesma. E isso pode ocorrer devido ao processo no qual a etapa de desenvolvimento está inserida ou por questões estratégicas da organização.

Já um processo de desenvolvimento de *software* para sistemas ERP possui peculiaridades que o diferem de outros sistemas. Dentre essas, algumas não podem ser modificadas pois não dependem da organização, como a necessidade das alterações legais para adequação à realidade fiscal. As alterações diárias que o sistema ERP costuma sofrer também são uma característica necessária devido a quantidade de clientes que o utilizam e a sua finalidade de contemplar diferentes módulos para servir como apoio à decisão gerencial.

Portanto a literatura possui um conceito definido com relação à Definição de Pronto que não considera algumas características de cada sistema. Com a disseminação do desenvolvimento ágil, em especial o *Scrum*, todos os tipos de organizações aplicam seus conceitos e não é diferente para empresas que desenvolvem ERP. E pode um time que desenvolve ERP, devido às suas características e ao tipo de sistema, ter uma Definição de Pronto que não segue à risca a teoria do *Scrum*?

Esse trabalho elaborou um estudo de caso em Times *Scrum* que desenvolvem um sistema ERP, avaliando o tempo de desenvolvimento e entrega de incrementos do produto a partir da Definição de Pronto que eles possuem. Para isso foram consideradas as peculiaridades que um sistema ERP possui e as questões culturais que envolvem a empresa estudada.

A partir da proposta do trabalho e das referências coletadas, esse capítulo detalha o estudo de caso realizado. Seu conteúdo inicia explicando o processo da empresa estudada, com suas características, segue apresentando os times escolhidos para a coleta dos dados e finaliza detalhando a análise de dados realizada, desde a preparação do *dataset* até a manipulação deles.

3.1 PROCESSO ATUAL DA EMPRESA DESENVOLVEDORA DO ERP

A organização estudada é uma empresa desenvolvedora de *software,* que possui um único sistema ERP e contempla diversas áreas. Os seus desenvolvedores são divididos em times de grandes áreas, sendo que dentro de cada área existe mais de um time, sendo cada um responsável por *n* funcionalidades.

Todos os clientes e usuários da empresa utilizam o sistema e, com isso, todos acabam sendo impactados pelas diversas alterações realizadas. Como o sistema é grande e altamente configurável, não são todos os clientes que utilizam as mesmas rotinas, sendo que algumas são utilizadas por um grupo pequeno de clientes e outras são utilizadas por quase todos.

Para diminuir o impacto gerado aos clientes, o sistema possui versões e *releases* que são criadas periodicamente. A organização possui um processo de geração semanal das implementações que foram produzidas e estão prontas, que consiste na liberação do produto nas demais versões e *releases*, sendo a maioria das gerações feitas nesse dia. Porém também há gerações nos outros dias para correções de erros identificados ou implementações que atendam demandas de legislações urgentes.

Algumas vezes as implementações e reestruturações de código fonte não são geradas imediatamente após sua conclusão, ficando na versão em que foi desenvolvida. Isso ocorre devido à complexidade da alteração e para garantir que chegue gradualmente para os clientes, passando por mais testes internos. Já na maior parte dos casos as alterações são geradas, mas não em todas as versões e *releases*, fazendo com que não atinja a todos os clientes de uma só vez.

As implementações podem ser de diferentes tipos, como mostra o Quadro 2. Esses tipos são definidos pelos técnicos responsáveis pelas demandas, no momento de sua criação, seguindo as definições da organização.

Quadro 2 - Tipos de demandas da empresa do estudo de caso

Tipo	Descrição das histórias
Verificação	Análises necessárias no sistema, para identificar se é um erro
Erro	*Bugs* no sistema
Legislação	Alterações legais necessárias no sistema
Impede processo	Alterações necessárias para liberar o funcionamento do sistema para um cliente
Saída específica	Alterações específicas para determinado cliente
Implementação	Implementações de funcionalidades no sistema
Infraestrutura	Alterações estruturais no sistema
Conversão	Alterações em tabelas do sistema que precisam ser convertidas
Requisito de projeto	Alterações referentes a projetos

Fonte: elaborado pelo autor

A empresa possui em seu processo algumas operações, que são etapas do processo, a serem realizadas para produção das demandas. Ela presume que pode haver problemas nessas operações que acabam refletindo no processo como um todo, visto que muitas vezes se tem a impressão de que algumas demandas ficam trancadas em determinadas etapas, o que acaba gerando um atraso para conclusão delas. O quadro 3 apresenta essas operações que iniciam no levantamento de requisitos das histórias com os clientes e vão até a geração das alterações realizadas para entrega ao cliente.

Quadro 3 - Tipos de operações a serem realizadas para produção de uma demanda

Tipo de operação	Descrição
Levantamento de requisitos	Levantamento dos requisitos realizado por técnico fora do time desenvolvimento, para obter as necessidades do cliente
Análise	Análise das histórias realizada pelo time desenvolvimento, para avaliar as alterações necessárias para atender o cliente
Acompanhamento/Instrução	Acompanhamento entre integrante mais experiente do time com integrante menos experiente, para disponibilizar ajuda técnica
Produção	Produção das alterações em código fonte realizada pelo time desenvolvimento
Revisão	Revisão do código fonte realizada pelo time desenvolvimento
Validação	Validação das histórias realizada por técnico fora do time desenvolvimento, para avaliar se o que foi produzido atende a necessidade do cliente
Geração	Geração das alterações do código fonte na versão/*release* necessária, realizada pelo time desenvolvimento após a história passar pela validação

Fonte: elaborado pelo autor

Os times da organização também possuem características em seus processos que são específicas e outras que são gerais entre todos os times. A metodologia *Scrum* também não é seguida à risca em alguns deles. Essas características são:

- As implementações podem ser quebradas em mais *Sprints* por um time, sendo realizadas sequencialmente e ficando concluídas ao final da última *Sprint*;
- As implementações podem ser divididas entre diferentes times e entre diferentes áreas, sendo que nesses casos elas são alinhadas para que sua produção ocorra em períodos próximos;
- As implementações costumam ter datas de previsão de entrega, que são organizadas pelo PO do time de acordo com as *Sprints*;

- A etapa de validação não depende do time para ser realizada, sendo realizada pelo responsável pela história e contemplando a verificação se o que foi feito atende o cliente final;
- Todos os times possuem uma etapa de refinamento dentro da *Sprint* atual, onde é feita a análise das próximas implementações;
- Todos os times possuem o processo de revisão do código fonte que é realizado por outro membro do time;
- Alguns times possuem um processo de testes de integração, que é realizado nas implementações após a revisão do código fonte;
- Nenhum time possui *review* em que participam os clientes e usuários finais da organização;
- Os times de infraestrutura também têm como seus clientes os demais times, devido às alterações que são feitas no *framework* interno. Com isso, nesses times o *review* é realizado com seus clientes;
- A maioria dos times possui demandas de erros a cada *Sprint*, que entram como não planejado e possuem prioridade para serem produzidos*;*
- Alguns times possuem demandas de legislação, que passam na frente de outras implementações previstas para as próximas *Sprints* e, por vezes, também entram como não planejado e com prioridade na *Sprint* corrente.

3.2 DESCRIÇÃO DOS CASOS PARA ESTUDO

Para seguir com o planejamento do estudo de caso, foram definidos os times dos quais foram coletados os dados, sendo que todos trabalham com desenvolvimento de *software*. Na Tabela 1 são apresentadas as características de cada um dos Times *Scrum* envolvidos neste estudo de caso.

Tabela 1 - Características gerais dos times de desenvolvimento do estudo de caso

Característica	Time #1	Time #2	Time #3	Time #4	Time #5
Quantidade de integrantes	5	3	5	5	3
Período de existência (meses)	48	48	48	48	36
Etapa considerada para DoD	Testes de integração	Validação	Revisão	Revisão	Validação

Fonte: elaborado pelo autor

Com relação à quantidade de integrantes do time, foi considerado o número ao final do período de coleta de dados. O período de existência do time levou em

consideração a quantidade de meses dentro do período de coleta de dados. A característica da Definição de Pronto do time diz respeito a etapa do processo que deve ser concluída para que o time considere a demanda como pronta.

Dada a caracterização geral dos times, no Quadro 4 são apresentadas algumas peculiaridades de cada time no que diz respeito às demandas que são atendidas, levantadas em análise pessoal do autor.

Quadro 4 - Peculiaridades dos times de desenvolvimento do estudo de caso

Peculiaridade	Time #1	Time #2	Time #3	Time #4	Time #5
Atende demandas de legislação	X				
Atende demandas de projeto				X	X
Atende demandas de infraestrutura				X	X
Atende demandas de implementações		X	X	X	
Possui demandas de erros	X		X	X	

Fonte: elaborado pelo autor

A partir da caracterização dos times, é possível perceber diferenças e semelhanças entre os Times *Scrum* participantes do estudo de caso. O tamanho dos times é bastante variado, assim como a sua respectiva Definição de Pronto. Também é possível perceber que todos os times são distintos no que diz respeito às suas peculiaridades, atendendo demandas que fazem ter cenários diferentes a cada *Sprint*. Esta variação nas características dos times colabora na abrangência do estudo de caso em questão.

Neste ponto, o estudo de caso incluiu a participação de um especialista, que possui atuação acadêmica relevante na área de Mineração de Dados. Foram realizadas reuniões para analisar a estrutura dos dados disponíveis e definir quais colunas seriam interessantes termos no *dataset,* considerando também o processo da empresa. Ficou definida a utilização da técnica de Árvore de Decisão para classificação dos dados e o *Jupyter Notebook*[1] como ferramenta para criação do modelo de *Machine Learning* desejado. Optou-se pela sua utilização ao invés do *Power BI*[2], como previsto inicialmente, pela possibilidade de geração de um modelo que gere *insights,* enquanto o anterior é um conjunto de ferramentas de BI mais focado

[1] https://jupyter.org/
[2] https://powerbi.microsoft.com/en-us/

para a visualização dos dados. A seguir, é apresentado o processo de Mineração de Dados segundo a literatura, indicando sua utilização no estudo de caso realizado.

3.3 MINERAÇÃO DE DADOS

A Mineração de Dados é um processo computacional para descobrir padrões em grandes conjuntos de dados, extraindo conhecimento de dados desorganizados (SANTANA, 2017). Conforme explica Santana (2020a), ela está relacionada à coleta, análise e interpretação de dados, possuindo as etapas para reunir os dados, transformá-los, analisá-los, interpretá-los e, por fim, comunicar as informações relevantes para a empresa. O objetivo é extrair dados de fontes de dados e transformá-los em informação útil, gerando valor para o negócio (SANTANA, 2017).

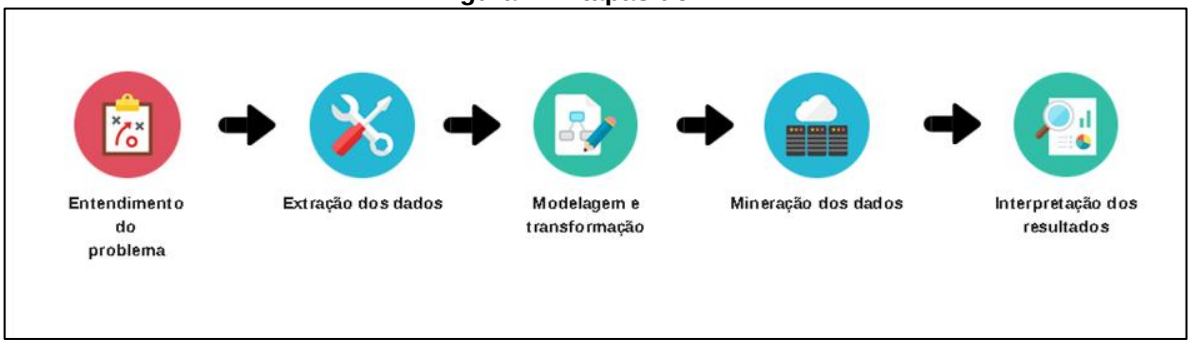
A Mineração de Dados, em geral, não analisa períodos, mas faz análises preditivas e constrói novas hipóteses. Após a coleta dos dados é necessário formular essas hipóteses que respondam aos problemas que estão sendo analisados, gerando informações e *insights* (SANTANA, 2020a). Através dos *insights* gerados podem ser tomadas decisões inteligentes para o negócio, como, por exemplo, predições de consumo, descoberta de padrões e comportamentos similares para uma melhor segmentação de clientes (SANTANA, 2019). A seguir serão detalhadas as etapas do processo de Mineração de Dados, a técnica de árvore de decisão e é explicada a tabela de matriz de confusão, utilizadas no presente trabalho.

3.3.1 DESCOBERTA DE CONHECIMENTO SOB BASE DE DADOS

A criação de inteligência de negócios sobre determinado assunto é o maior benefício da Mineração de Dados (SANTANA, 2017), e para criar essa inteligência é necessário que seja feita a análise de grandes quantidades de dados. Como explica Goldschmidt (2005), é imprescindível o desenvolvimento de ferramentas que auxiliem o homem, de forma automática e inteligente, na tarefa de analisar, interpretar e relacionar esses dados para que se possa desenvolver e selecionar estratégias de ação em cada contexto de aplicação.

Para atender esse contexto existe a *Knowledge Discovery in Databases* (KDD), ou Descoberta de Conhecimento sob Base de Dados, como apresentado nessa seção.

Figura 4 - Etapas do KDD



Fonte: Santana (2017)

A Figura 4 mostra as etapas da KDD. Na etapa de entendimento do problema é realizada a compreensão do problema, bem como o escopo e os objetivos do projeto. É muito importante que se tenha um bom conhecimento do negócio, para que fiquem claros quais os benefícios esperados da solução e seus principais desafios. Já na extração dos dados são realizadas ações de extração e coleta de dados, identificando as bases de dados que serão utilizadas no projeto bem como suas tabelas e atributos relevantes. Essa coleta de dados deve ser realizada a partir de sistemas locais e, dependendo da solução, também de outras fontes como a internet, centralizando e agrupando os dados necessários (SANTANA, 2017).

A modelagem, conforme explica Goldschmidt (2005), possui fundamental relevância no processo de descoberta de conhecimento, compreendendo desde a correção de dados errados até o ajuste da formatação dos dados para os algoritmos de Mineração de Dados a serem utilizados. Aqui é feita a seleção de atributos relevantes, limpeza de dados inconsistentes, detecção e tratamento de anomalias, conversão e transformação de tipos de dados etc. Uma base de dados bem modelada influencia significativamente na performance e eficiência dos modelos.

É na etapa de mineração onde é definido o método de Mineração de Dados, consistindo em definir a melhor técnica a ser aplicada e a abordagem para a melhor execução dos algoritmos (SANTANA, 2017). Como explica Goldschmidt (2005), é a principal etapa do processo de KDD porque ocorre a busca efetiva por conhecimentos novos e úteis a partir dos dados. Por fim, a interpretação dos resultados faz a avaliação e interpretação dos resultados de todo o processo. É importante que o conhecimento gerado seja documentado e utilizado para a otimização dos processos com o objetivo de agregar valor ao negócio (SANTANA, 2017).

3.3.2 ÁRVORE DE DECISÃO

Árvore de decisão é uma das técnicas mais populares de Mineração de Dados (SILVA; PERES; BOSCARIOLI, 2016). Segundo Santana (2020b), é um tipo de algoritmo de aprendizagem de máquina supervisionado que se baseia na ideia de divisão dos dados em grupos homogêneos, tendo como objetivo encontrar o atributo que gera a melhor divisão dos dados. Como explicam Silva, Peres e Boscarioli (2016), ela consiste em uma coleção de nós internos e nós folhas, organizados em um modelo hierárquico.

Uma árvore de decisão usa uma estrutura de árvore para representar um número de possíveis caminhos de decisão e um resultado para cada caminho. Ela possui muitas recomendações e são muito fáceis de entender e interpretar, sendo que o processo por onde chegam em uma previsão é completamente transparente (GRUS, 2016).

Algumas características, segundo Santana (2020b):

- Possui um fácil entendimento: não requer nenhum conhecimento estatístico para a sua interpretação;
- Aceita tanto dados categóricos quanto numéricos: diminui a necessidade da limpeza de dados em comparação com outros modelos;
- É propensa a sofrer *overfitting*: ao se ajustar muito aos dados de treino é possível que não tenha um desempenho muito bom com os dados de teste;
- São instáveis: pequenas alterações nos dados de treino produzem novas árvores.

As árvores de decisão podem ser utilizadas em um cenário de classificação ou regressão (SANTANA, 2020b). A classificação, como explicam Silva, Peres e Boscarioli (2016), é o processo pelo qual se determina um mapeamento capaz de indicar a qual classe pertence qualquer exemplar de um domínio sob análise, com base em um conjunto de dados já classificado. Já a regressão é usada para estimar valores a partir de um conjunto de dados histórico.

Ao gerar uma árvore de decisão a partir de uma base que tenha um grande número de atributos, e se alguns critérios não forem especificados a fim de controlar a árvore, corre-se o risco de ela representar cada amostra de dados, gerando uma árvore muito grande e um modelo extremamente complexo. Nesse caso, ela terá bons

resultados para os dados de treino e baixo desempenho para os dados de teste, ou seja, vai sofrer *overfitting*.

Como explica Santana (2020b), algumas estratégias podem ser adotadas para ajudar a evitar o *overfitting,* como:

- Especificar o número mínimo de amostras para divisão do nó;
- Especificar o número mínimo de amostras para o nível folha;
- Definir a profundidade máxima da árvore;
- Definir o número máximo de *features* para considerar durante a divisão.

3.3.3 MATRIZ DE CONFUSÃO

Uma matriz de confusão é uma tabela que indica os erros e acertos do seu modelo, comparando com o resultado esperado (RODRIGUES, 2019).

Figura 5 - Demonstração de uma matriz de confusão

		Detectada	
		Sim	**Não**
Real	**Sim**	Verdadeiro Positivo (VP)	Falso Negativo (FN)
	Não	Falso Positivo (FP)	Verdadeiro Negativo (VN)

Fonte: Rodrigues (2019)

A Figura 5 demonstra uma matriz de confusão, podendo ser seus dados:

- Verdadeiro positivo: ocorre quando, no conjunto real, a classe que está se buscando foi prevista corretamente.
- Falso positivo: ocorre quando, no conjunto real, a classe que está se buscando prever foi prevista incorretamente.
- Verdadeiro negativo: ocorre quando, no conjunto real, a classe que não se está buscando prever foi prevista corretamente.
- Falso negativo: ocorre quando, no conjunto real, a classe que não está se buscando prever foi prevista incorretamente.

Ela é uma matriz de valores reais e valores preditos pelo seu classificador. Exibir a matriz de confusão é uma forma intuitiva de saber como seu classificador está se comportando (SANTANA, 2018).

Ao ser feita a contagem de todos os termos e obter a matriz de confusão, é possível calcular métricas de avaliação para a classificação. Como explica Rodrigues (2019), essas métricas podem ser:

- Acurácia: indica uma performance geral do modelo. Dentre todas as classificações, quantas o modelo classificou corretamente;
- Precisão: dentre todas as classificações de classe positivo que o modelo fez, quantas estão corretas;
- *Recall* / Sensibilidade: dentre todas as situações de classe positivo como valor esperado, quantas estão corretas;
- *F1-Score*: média harmônica entre precisão e *recall*.

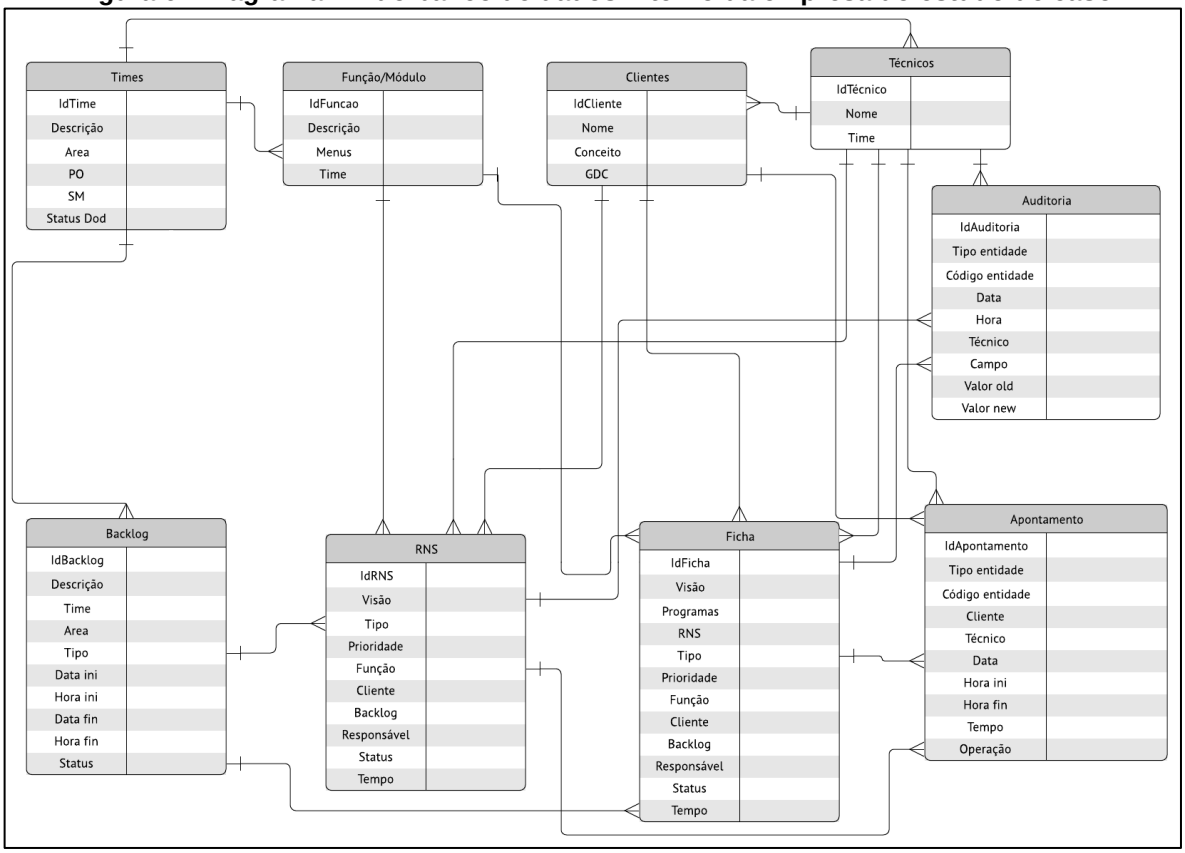
Concluída a apresentação dos conceitos de Mineração de Dados, a seguir será detalhada a preparação do *dataset* e a análise realizada nos dados.

3.4 PREPARAÇÃO DO *DATASET*

Para ser realizada a coleta de dados, a primeira questão a ser identificada é a origem de onde os dados serão extraídos. No presente trabalho os dados vieram do banco de dados do *Customer Relationship Management* (CRM) interno da empresa, que disponibiliza os apontamentos realizados pelos integrantes dos times, assim como demais colaboradores, nas demandas que são realizadas. Esses apontamentos são registros de atividades que contêm informações sobre às mesmas, como data/hora inicial, data/hora final, tipo de operação realizado, estado do processo que a demanda estava e para qual estado foi após o apontamento.

Esse banco de dados é composto de diversas tabelas, como mostra o diagrama ER na Figura 6, onde estão separados os dados dos times, clientes, *backlogs*, demandas, apontamentos etc.

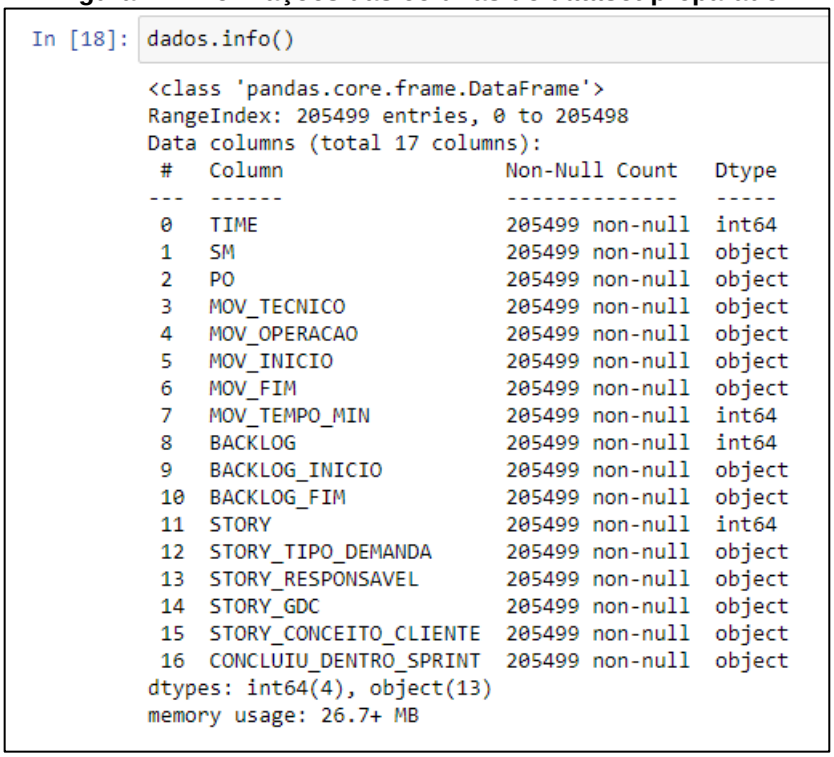
Figura 6 - Diagrama ER do banco de dados interno da empresa do estudo de caso



Fonte: elaborado pelo autor

Finalizamos a versão inicial do *dataset* com 17 colunas, conforme mostra a Figura 7, retirada do programa em que foi realizada a análise, possuindo mais de 200 mil linhas de dados entre todos os times, o que consideramos um número adequado de dados para a análise que precisa de uma grande quantidade de informações para abranger várias situações.

Figura 7 - Informações das colunas do *dataset* preparado



```
In [18]: dados.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205499 entries, 0 to 205498
Data columns (total 17 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   TIME                    205499 non-null  int64
 1   SM                      205499 non-null  object
 2   PO                      205499 non-null  object
 3   MOV_TECNICO             205499 non-null  object
 4   MOV_OPERACAO            205499 non-null  object
 5   MOV_INICIO              205499 non-null  object
 6   MOV_FIM                 205499 non-null  object
 7   MOV_TEMPO_MIN           205499 non-null  int64
 8   BACKLOG                 205499 non-null  int64
 9   BACKLOG_INICIO          205499 non-null  object
 10  BACKLOG_FIM             205499 non-null  object
 11  STORY                   205499 non-null  int64
 12  STORY_TIPO_DEMANDA      205499 non-null  object
 13  STORY_RESPONSAVEL       205499 non-null  object
 14  STORY_GDC               205499 non-null  object
 15  STORY_CONCEITO_CLIENTE  205499 non-null  object
 16  CONCLUIU_DENTRO_SPRINT  205499 non-null  object
dtypes: int64(4), object(13)
memory usage: 26.7+ MB
```

Fonte: capturado pelo autor

O quadro 5 explica cada coluna coletada, bem como sua descrição e sua referência na tabela do banco de dados.

Quadro 5 - Detalhamento das colunas preparadas no *dataset*

(continua)

Coluna (Referência no banco de dados)	Descrição
TIME (Times)	Time que pertence o apontamento realizado
SM (Técnicos)	SM do time
PO (Técnicos)	PO do time
MOV_TECNICO (Técnicos)	Técnico que realizou o apontamento
MOVE_OPERACAO (Apontamento)	Tipo de operação que foi realizada no apontamento
MOV_INICIO (Apontamento)	Data/hora inicial do apontamento
MOV_FIM (Apontamento)	Data/hora final do apontamento
MOV_TEMPO_MIN (Apontamento)	Tempo de duração do apontamento em minutos
BACKLOG (Backlog)	*Backlog* no qual pertence a história que foi executada

Quadro 5 – Detalhamento das colunas preparadas no dataset

(conclusão)

Coluna (Referência no banco de dados)	Descrição
BACKLOG_INICIO (Backlog)	Data/hora inicial do *backlog*
BACKLOG_FIM (Backlog)	Data/hora final do *backlog*
STORY (RNS / Ficha)	ID da história que foi executada
STORY_TIPO_DEMANDA (RNS / Ficha)	Tipo da história que foi executada
STORY_RESPONSAVEL (RNS / Ficha)	Responsável da história que foi executada
STORY_GDC (RNS / Ficha)	Gerente de contas do cliente que solicitou a história
STORY_CONCEITO_CLIENTE (RNS / Ficha)	Conceito do cliente que solicitou a história para a empresa
CONCLUIU_DENTRO_SPRINT (DoD)	Coluna que indica se o apontamento foi realizado dentro da *Sprint*

Fonte: elaborado pelo autor

Como o objetivo do estudo é avaliar a Definição de Pronto dos times, a principal informação do *dataset* preparado é a indicação dessa definição. A coluna principal do *dataset* é a “CONCLUIU_DENTRO_SPRINT” e indica se o apontamento foi realizado dentro da *Sprint*, considerando a data/hora final do *backlog* ao qual a história pertence. Inicialmente essa coluna tinha como objetivo considerar se o apontamento havia atendido a Definição de Pronto de cada time, porém isso foi alterado devido à pouca relevância que ela teria considerando as seguintes situações:

- A DoD dos times pode ter mudado durante o período analisado, porém não chegou perto do ideal, que seria entregue ao cliente final;
- Não iria considerar algumas etapas do processo que são mais próximas da entrega final, como as operações de validação e geração;
- Essa coluna só teria a informação de quantas histórias os times não conseguem concluir o desenvolvimento, que poderia gerar uma análise superficial.

A DoD de cada time praticamente estagnou desde a implantação do *Scrum*. Idealmente deveria considerar esta definição no início e haver uma evolução com o tempo. Devido a isso, pegou-se o cenário ideal de desenvolvimento, que seria a execução de todas as operações do processo da empresa e entrega para o cliente

final, verificando quais etapas que não foram realizadas dentro da *Sprint* do time, ou seja, que falham mais.

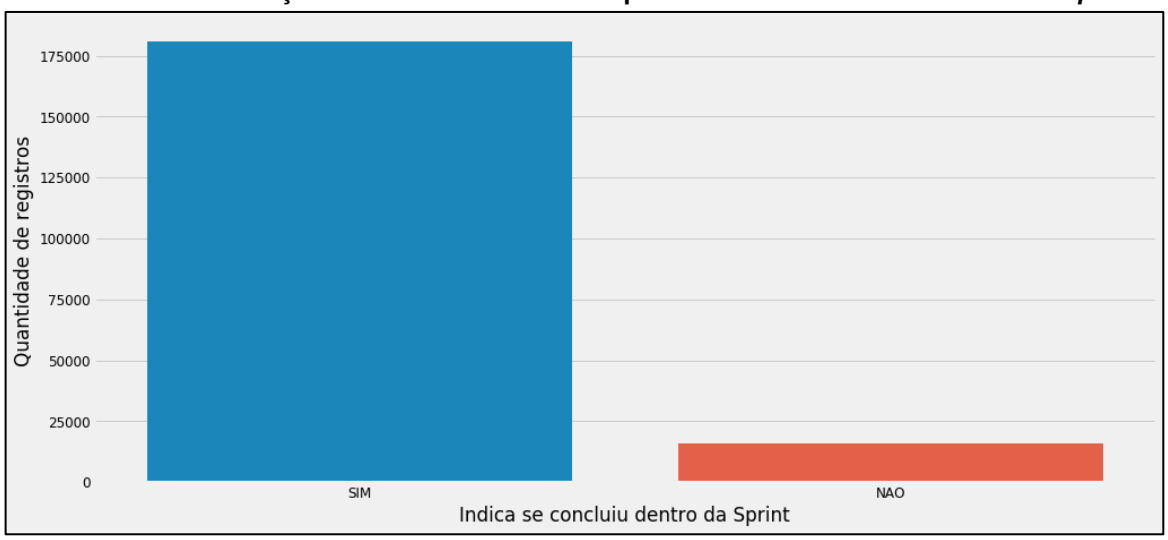
A implantação do *Scrum* na empresa foi realizada a partir de 2015, sendo que durante o ano foram definidos os times e como seriam suas estruturas. Considerando que esse período inicial foi de consolidação da metodologia na empresa, foram coletados todos os dados entre os anos de 2016 e 2019, com os quatro anos completos. Desta forma, foram ignorados os dados durante o período inicial e foram considerados somente a partir do momento que a metodologia já estava consolidada. Também fechou um ciclo de análise, onde podem ser desconsideradas divergências devido à período de férias na empresa.

Com relação às histórias, somente foram considerados apontamentos que pertencem à *Sprint Backlog* dos times, ou seja, considerou-se o que foi planejado pelo time e não foram examinadas as demandas urgentes de cada *Sprint*. Isso foi feito porque o encaminhamento de demandas pelos times já considera que o tempo não é todo alocado na *Sprint* e que haverá erros. Além disso, demandas urgentes são consideradas como prioritárias e que serão cumpridas, na sua grande maioria, dentro da *Sprint*.

3.5 COLETA E ANÁLISE DOS DADOS

A partir do *dataset* preparado iniciaram-se as análises exploratórias. A primeira coluna em que se realizou a distribuição dos dados, para identificar as maiores ocorrências, foi a que indica conclusão dentro da *Sprint*, conforme mostra o Gráfico 1.

Gráfico 1 - Distribuição dos dados do *dataset* que indicam se concluiu dentro da *Sprint*

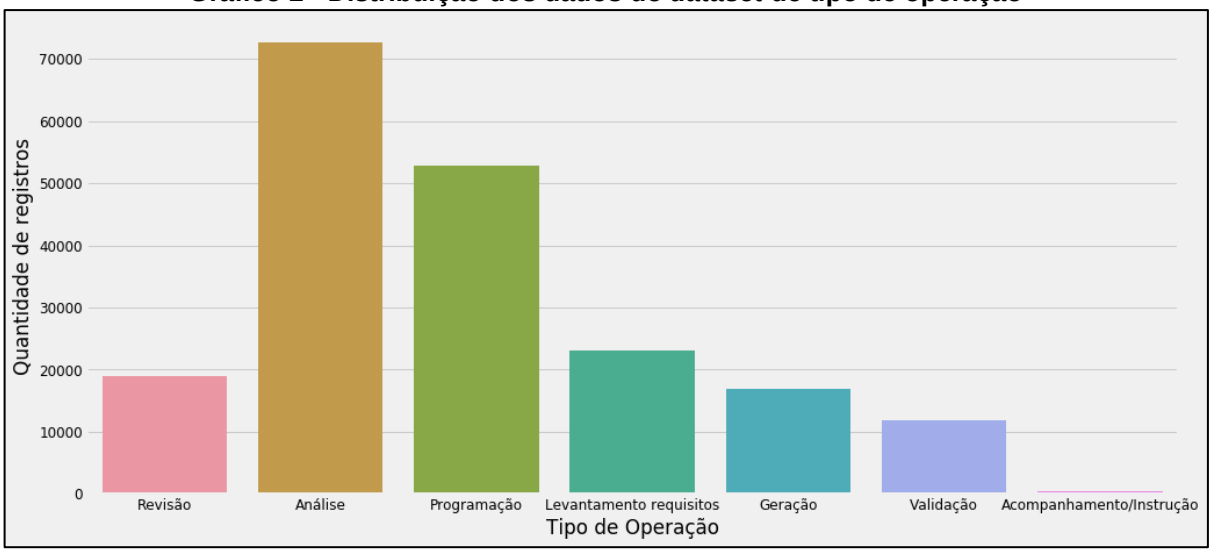


Fonte: elaborado pelo autor

Com essa distribuição foi possível perceber que a grande maioria dos apontamentos foram realizados dentro da *Sprint*, o que faz sentido, uma vez que estão sendo analisados apontamentos de todas as etapas do processo, incluindo o levantamento de requisitos e análise, que são executados, inclusive, antes do início de cada *Sprint*.

Também foi feita a distribuição dos dados da coluna de tipo de operação, conforme mostra o Gráfico 2, para identificar quais são os mais presentes.

Gráfico 2 - Distribuição dos dados do *dataset* de tipo de operação

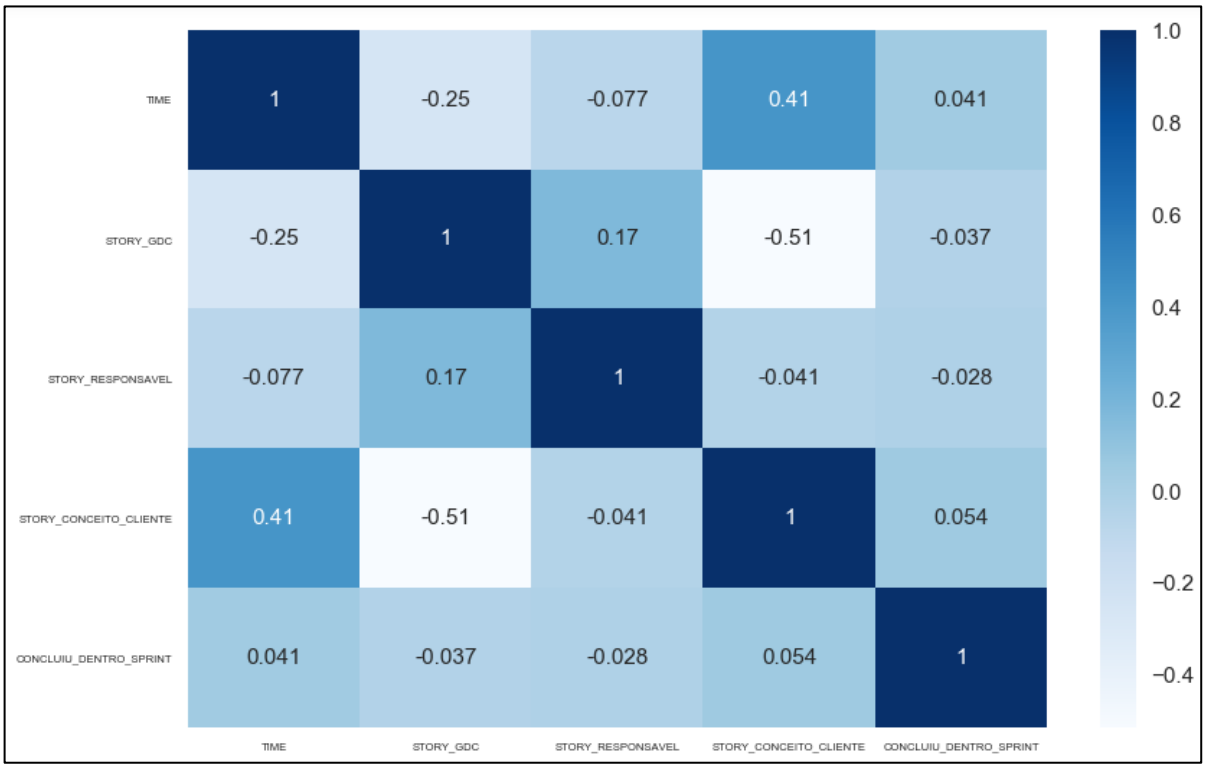


Fonte: elaborado pelo autor

Através da análise do gráfico foi possível perceber que a distribuição dos dados se concentra com maior quantidade nos registros de análise e programação, que são apontamentos realizados pelo time desenvolvimento e não envolvem nenhum outro setor.

Para verificar se a conclusão dentro da *Sprint* era influenciada por alguma característica das histórias, foi feita uma correlação entre esses dados como mostra o Gráfico 3.

Gráfico 3 - Correlação dos dados que indicam conclusão dentro da *Sprint* com as características das histórias



Fonte: elaborado pelo autor

Como mostra o gráfico, nenhum dos dados teve uma correlação forte com outro. Quanto mais próximo de 1, ou azul escuro no gráfico, mais forte é a correlação positiva, indicando que dois dados aumentam juntos o número de suas ocorrências. Já quanto mais próximo de -1, ou branco no gráfico, mais forte é a correlação negativa, indicando que dois dados se relacionam de forma que enquanto um aumenta e outro diminuí o número de suas ocorrências.

Após a etapa de análises exploratórias, com a participação do especialista, foram levantados os melhores caminhos a seguir para a análise e construção do modelo de *Machine Learning*. Para isso, procurou-se identificar quais eram os atributos relevantes e a correlação entre todos os dados.

Foi visto que havia algumas colunas no *dataset* que eram referentes a dados categóricos e que, por serem dados que mudam a cada *Sprint,* não valeriam para todo o período. Com isso, as colunas do *backlog*, data/hora inicial e final do *backlog*, ID da história, data/hora inicial e final do apontamento e tempo de duração do apontamento foram eliminadas a cada análise, dentro da ferramenta, conforme mostra a Figura 8.

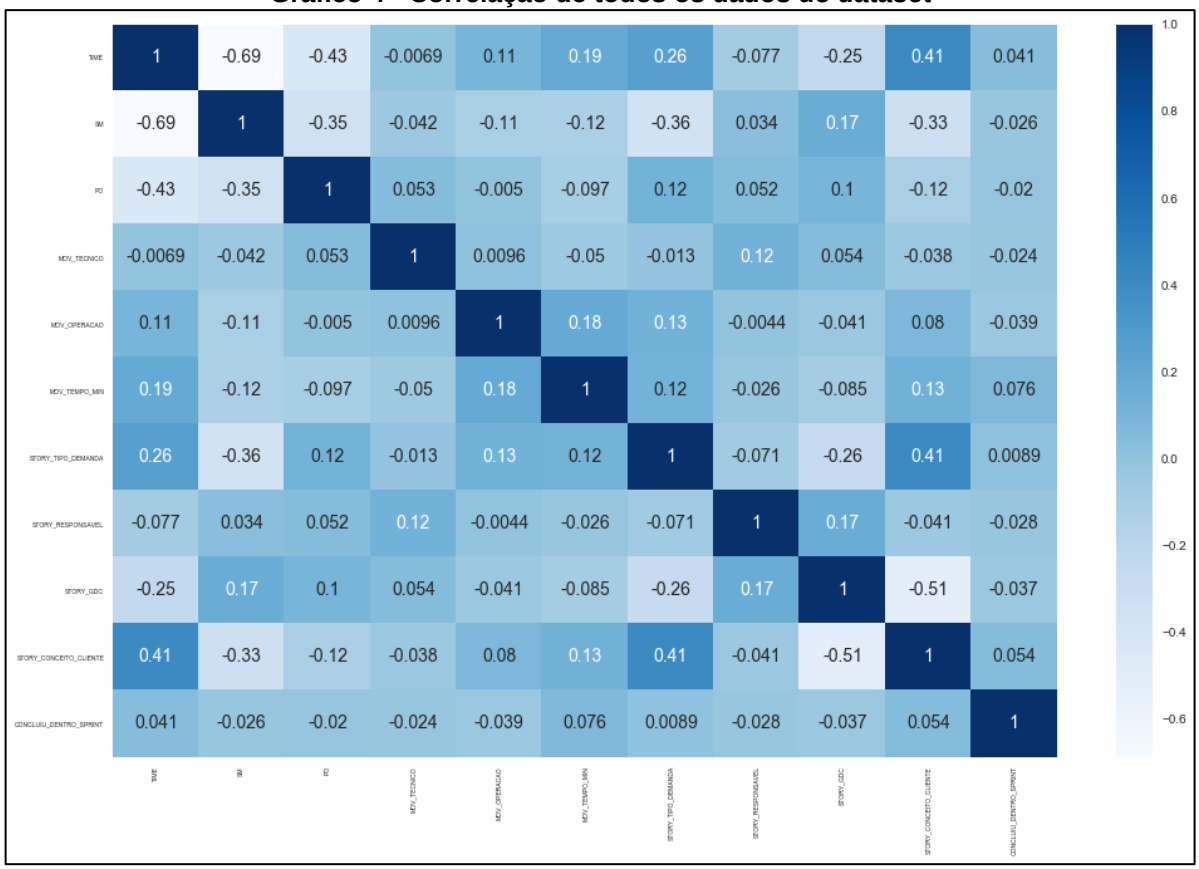
Figura 8 - Comandos para eliminação de colunas do *dataset*

```
In [78]: dados.drop('MOV_INICIO', axis=1, inplace=True)
         dados.drop('MOV_FIM', axis=1, inplace=True)
         dados.drop('MOV_TEMPO_MIN', axis=1, inplace=True)
         dados.drop('BACKLOG', axis=1, inplace=True)
         dados.drop('BACKLOG_INICIO', axis=1, inplace=True)
         dados.drop('BACKLOG_FIM', axis=1, inplace=True)
         dados.drop('STORY', axis=1, inplace=True)
```

Fonte: elaborado pelo autor

Após realizada limpeza no *dataset* para a análise, foi refeita a correlação entre todas as colunas disponíveis para identificar possíveis padrões existentes. O Gráfico 4 mostra a correlação de todos os dados.

Gráfico 4 - Correlação de todos os dados do *dataset*



Fonte: elaborado pelo autor

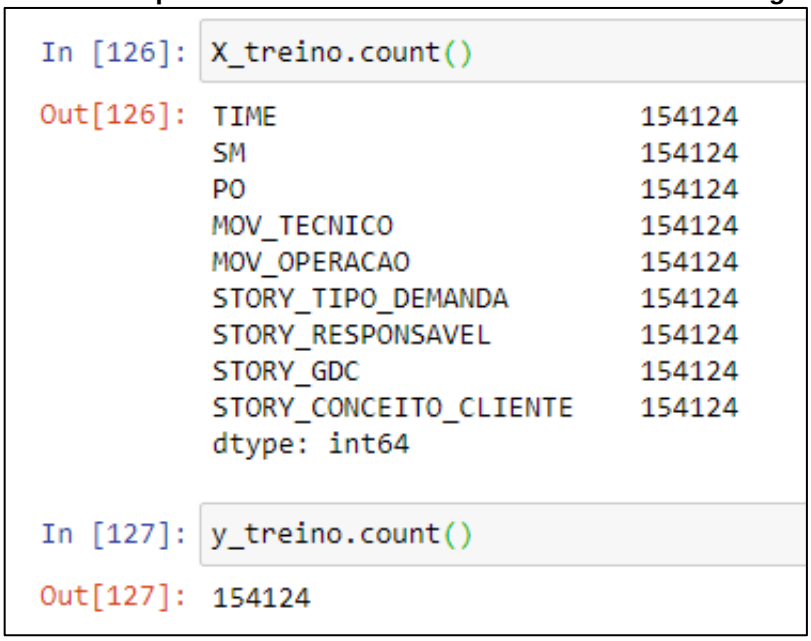
Nessa nova análise identificou-se que as correlações mais fortes são as mais óbvias, como entre time e SM, e time e PO. Outra observação interessante é a relação entre as colunas tipo de demanda com o time e o SM, o que ocorre devido às demandas que cada time mais produz e indica ser um comportamento do ERP.

Também foi visto que existe uma correlação maior entre o tipo de demanda com o conceito do cliente, indicando que os clientes com conceitos maiores tendem a ter mais implementações e projetos, ou que demandas de infraestrutura tendem a serem feitas mais como demandas internas.

Porém a coluna principal de análise, que indica se concluiu dentro da *Sprint*, não teve nenhuma relação considerável com outra coluna, o que mostra não existir nenhum padrão claro.

Seguindo para a construção do modelo de *Machine Learning*, ele foi aplicado com árvore de decisão. Para isso, foi feita a separação da classe “CONCLUIU_DENTRO_SPRINT” para treino e teste, ficando 75% dos seus dados para o primeiro e 25% para o segundo, como mostram as Figuras 9 e 10.

Figura 9 - Comandos para treinamento do modelo de *Machine Learning* no *dataset*



Fonte: elaborado pelo autor

Figura 10 - Comandos para teste do modelo de *Machine Learning* no *dataset*

```
In [130]: X_teste.count()

Out[130]: TIME                    51375
          SM                      51375
          PO                      51375
          MOV_TECNICO             51375
          MOV_OPERACAO            51375
          STORY_TIPO_DEMANDA      51375
          STORY_RESPONSAVEL       51375
          STORY_GDC               51375
          STORY_CONCEITO_CLIENTE  51375
          dtype: int64

In [131]: y_teste.count()

Out[131]: 51375
```

Fonte: elaborado pelo autor

Foi feita validação do modelo e verificou-se, através da matriz de confusão gerada, que atingiu um índice de 94% de precisão geral, como mostra a Figura 11, o que pode ser considerado um índice satisfatório porque vai acertar a maior parte das simulações para indicar se o apontamento concluiu dentro da *Sprint*. Porém dentre os casos que indicam que concluiu dentro da *Sprint*, indicado pelo valor “1”, a precisão ficou em 96%, já nos casos que indicam que não concluiu dentro da *Sprint,* indicado pelo valor “0”, foi apenas de 68%, sendo que esses são os casos que mais importantes para o estudo.

Figura 11 - Índices do modelo de *Machine Learning* do *dataset* completo

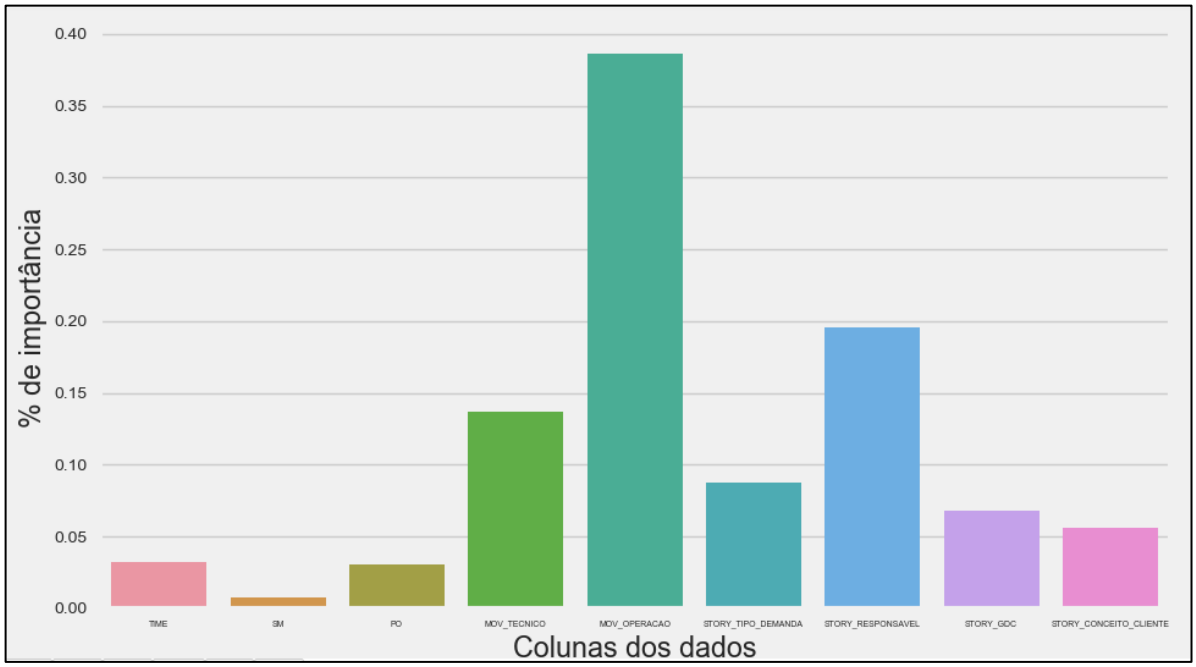
```
In [139]: from sklearn import metrics
          print(metrics.classification_report(y_teste,resultado))

                        precision    recall  f1-score   support

                     0       0.68      0.59      0.63      4157
                     1       0.96      0.98      0.97     47218

              accuracy                           0.94     51375
             macro avg       0.82      0.78      0.80     51375
          weighted avg       0.94      0.94      0.94     51375
```

Fonte: elaborado pelo autor

No modelo de *Machine Learning* criado, foi feito levantamento das colunas que tiveram maior importância, como apresentado no Gráfico 5. É possível perceber que

o tipo de operação é a coluna com maior importância para definição se conclui ou não dentro da *Sprint*.

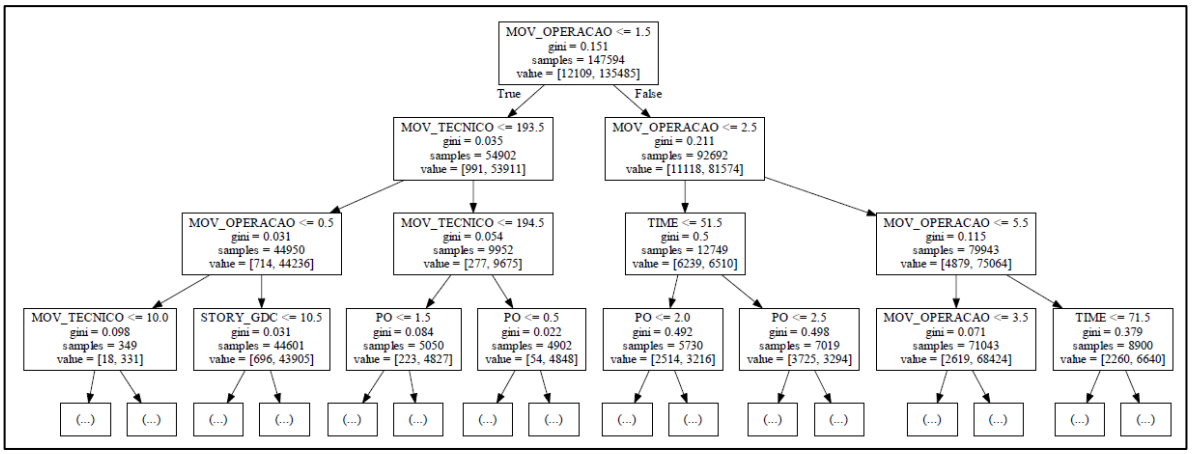
Gráfico 5 - Importância das *features* no modelo de *Machine Learning* do *dataset* completo



Fonte: elaborado pelo autor

Analisando a Árvore de Decisão gerada pelo modelo, apresentada na Figura 12, o tipo de operação, que é uma informação do processo, também aparece como mais relevante, seguido do técnico que realizou o apontamento e o time, que são informações mais relacionadas às pessoas.

Figura 12 - Árvore de decisão gerada pelo modelo de *Machine Learning* do *dataset* completo



Fonte: elaborado pelo autor

Com relação a operação, a linha de análise apresentada pela árvore mostra as seguintes relações de acordo com o tipo da operação:

- Quando é análise: O responsável da história e o técnico que realizou o apontamento se mostram mais relevantes;
- Quando é programação: O tipo de demanda e o time são os mais relevantes, mostrando que há diferenças entre eles;
- Quando é revisão: O técnico que realizou o apontamento é o mais relevante;
- Quando e validação: O responsável da história é o mais relevante;
- Quando é geração: O time é o mais relevante.

Como a operação se mostrou a coluna mais relevante para definir se a demanda é concluída dentro da *Sprint*, foi realizada uma distribuição de frequências das colunas “MOV_OPERACAO” e “CONCLUIU_DENTRO_SPRINT. A Tabela 2 mostra de forma detalhada quantos casos foram concluídos e quantos não foram concluídos dentro da *Sprint* para cada operação.

Tabela 2 - Estatística de apontamento com conclusão dentro da *Sprint* por operação utilizados no estudo de caso

Operação	Concluiu na *Sprint* = Sim	Concluiu na *Sprint* = Não
Levantamento de requisitos	92%	8%
Análise	98%	2%
Programação	98%	2%
Revisão	96%	4%
Validação	75%	25%
Geração	52%	48%

Fonte: elaborado pelo autor

Além da distribuição de frequências, também foi realizado um teste removendo a coluna de operação, para verificar se iriam aparecer outra *features* mais impactantes. Desta forma, de um modo geral, a precisão do modelo gerado baixou, principalmente quando buscou identificar o que não concluiu dentro da *Sprint*, como mostra a Figura 13, fazendo com que o modelo não tivesse um bom resultado.

Figura 13 - Índices do modelo de *Machine Learning* do *dataset* removendo a coluna de operação

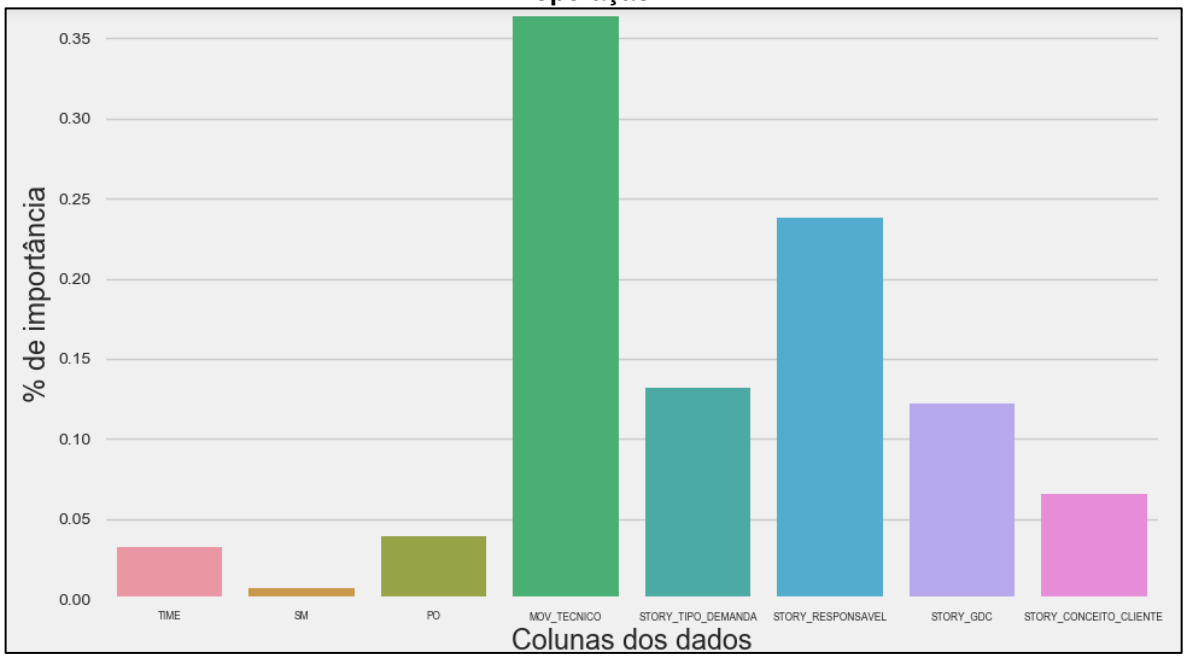
```
In [40]: from sklearn import metrics
         print(metrics.classification_report(y_teste,resultado))
```

	precision	recall	f1-score	support
0	0.57	0.19	0.29	4053
1	0.93	0.99	0.96	45146
accuracy			0.92	49199
macro avg	0.75	0.59	0.62	49199
weighted avg	0.90	0.92	0.90	49199

Fonte: elaborado pelo autor

Não foi possível montar um modelo que identifique quando não vai concluir dentro da *Sprint* sem a informação da operação, pois as demais informações sozinhas se mostraram menos relevantes para identificar a conclusão ou não da demanda dentro da *Sprint*. Mesmo assim, dentre elas o responsável da história e o técnico que realizou o apontamento, que são informações relacionadas às pessoas, se mostraram mais relevantes, como mostra o Gráfico 6.

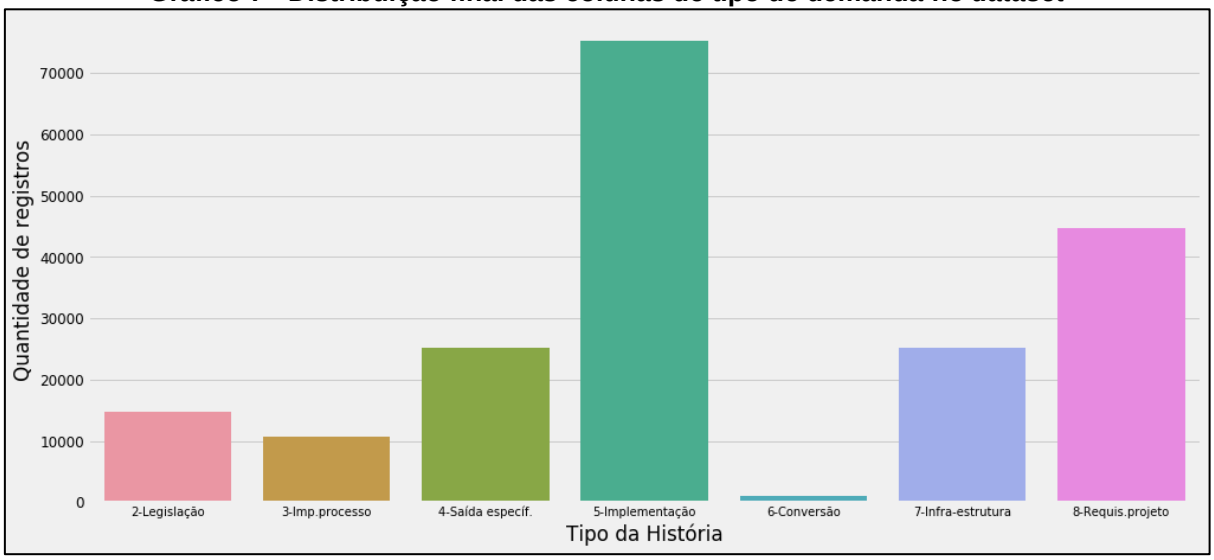
Gráfico 6 - Importância das *features* no modelo de *Machine Learning* removendo a coluna de operação



Fonte: elaborado pelo autor

Após análise das árvores de decisão geradas e da distribuição de frequência das colunas, foram vistas algumas mudanças a serem realizadas no *dataset* para ser refeito o processamento dos dados. Foram removidos os apontamentos com tipo de demanda “erros“ e “verificações” mesmo quando planejados na *Sprint*, porque acabam impactando nas operações de programação. Com a análise realizada, se comprovou de fato que os erros são prioritários com relação as demais demandas, o que podemos considerar como uma característica do sistema ERP. O Gráfico 7 mostra a distribuição final das colunas de tipo de demanda.

Gráfico 7 - Distribuição final das colunas de tipo de demanda no *dataset*



Fonte: elaborado pelo autor

Para aprofundar a análise focando nas operações, foi alterado o *dataset* para conter somente os apontamentos referentes às operações de “programação” e “revisão”, que são competências do time desenvolvimento, aplicando o modelo de Mineração dos Dados. Com os resultados foi possível identificar que, conforme mostra a Figura 14, a precisão geral aumentou uma vez que o modelo foi assertivo quase a totalidade dos casos para definir quando conclui dentro da *Sprint*, porém ficou com um percentual muito baixo na precisão para definir quando não concluiu dentro da *Sprint,* devido ao pequeno número de casos que não concluiu dentro da *Sprint* nessas operações.

Figura 14 - Índices do modelo de *Machine Learning* do *dataset* removendo as operações de programação e revisão

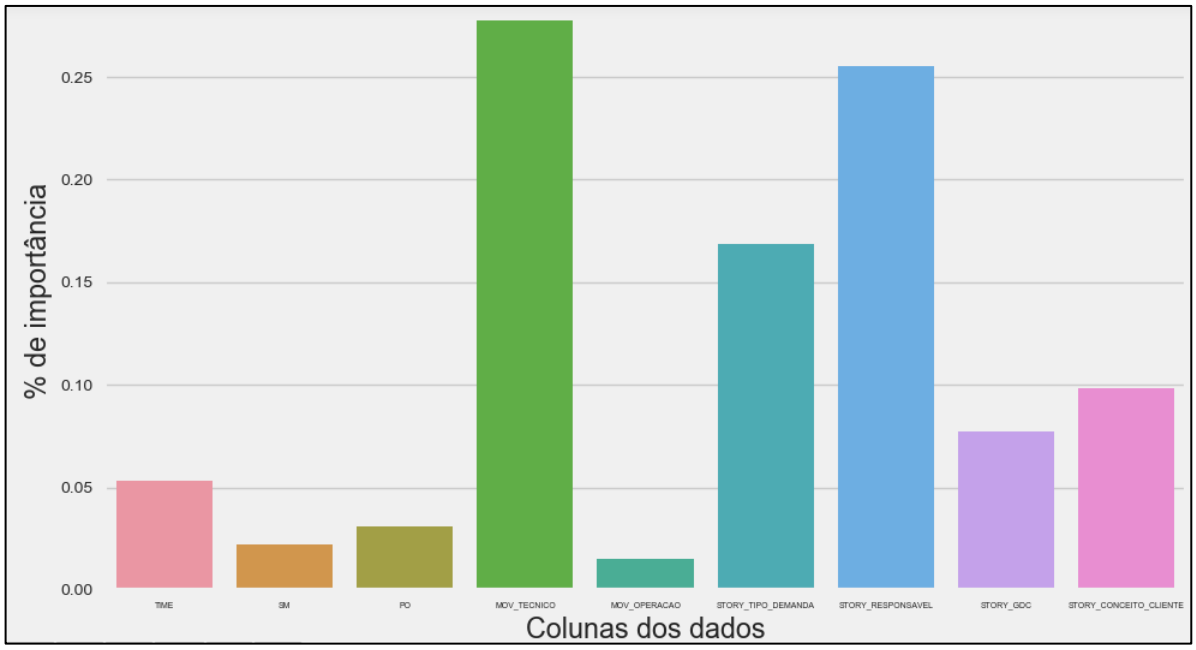
```
In [40]: from sklearn import metrics
         print(metrics.classification_report(y_teste,resultado))
```

	precision	recall	f1-score	support
0	0.49	0.17	0.25	399
1	0.98	1.00	0.99	17518
accuracy			0.98	17917
macro avg	0.73	0.58	0.62	17917
weighted avg	0.97	0.98	0.97	17917

Fonte: elaborado pelo autor

Dentre as *features* mais relevantes para esse cenário, novamente o técnico que realizou o apontamento e o responsável da história aparecem, como mostra o Gráfico 8. O mesmo comportamento também foi observado na árvore de decisão gerada.

Gráfico 8 - Importância das *features* no modelo de *Machine Learning* do *dataset* removendo as operações de programação e revisão



Fonte: elaborado pelo autor

Seguindo a mesma linha da análise anterior, foi alterado o *dataset* para conter somente os apontamentos referentes às operações de “validação” e “geração”, que não são competências do time desenvolvimento. Essa análise foi realizada com o intuito de identificar os problemas reais, uma vez que a quantidade de casos que concluem e não concluem dentro da *Sprint* são bem próximos nessas operações.

Como mostra a Figura 15, a precisão geral baixou em relação ao modelo inicial, porém aumentou consideravelmente a precisão quando não concluiu dentro da *Sprint*, que é o mais importante, se mostrando o modelo mais confiável.

Figura 15 - Índices do modelo de *Machine Learning* do *dataset* removendo as operações de validação e geração

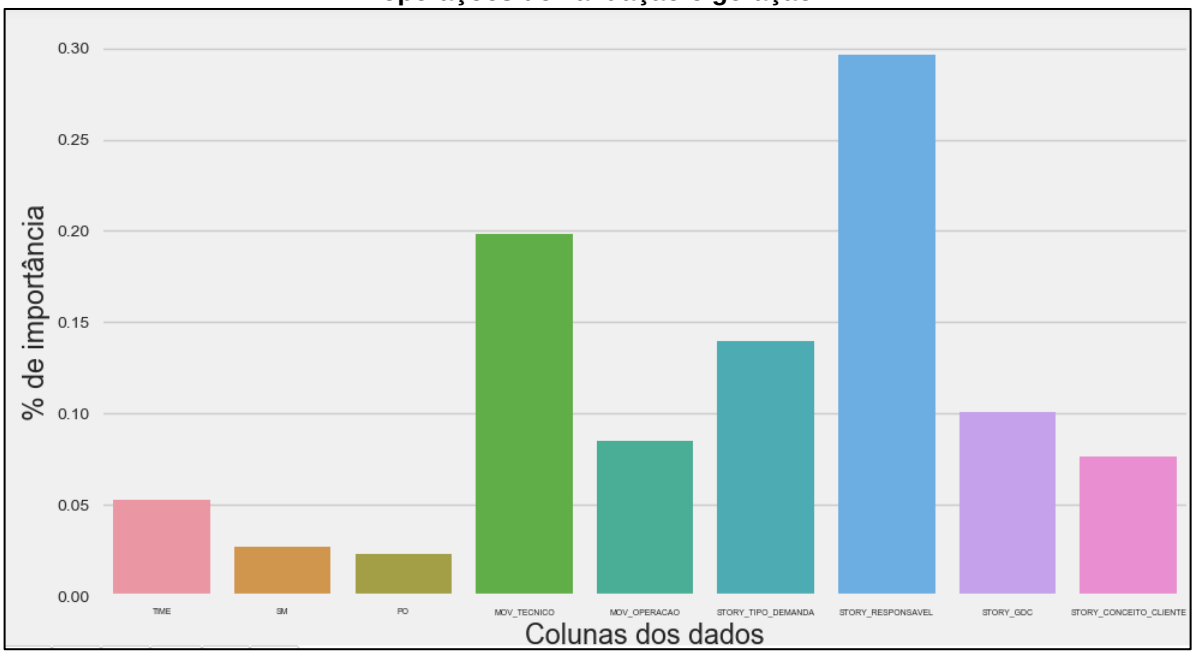
```
In [40]: from sklearn import metrics
         print(metrics.classification_report(y_teste,resultado))
```

	precision	recall	f1-score	support
0	0.73	0.74	0.73	2816
1	0.83	0.82	0.83	4387
accuracy			0.79	7203
macro avg	0.78	0.78	0.78	7203
weighted avg	0.79	0.79	0.79	7203

Fonte: elaborado pelo autor

Analisando as colunas mais relevantes, nesse cenário o responsável da história se mostrou como primeiro, conforme Gráfico 9.

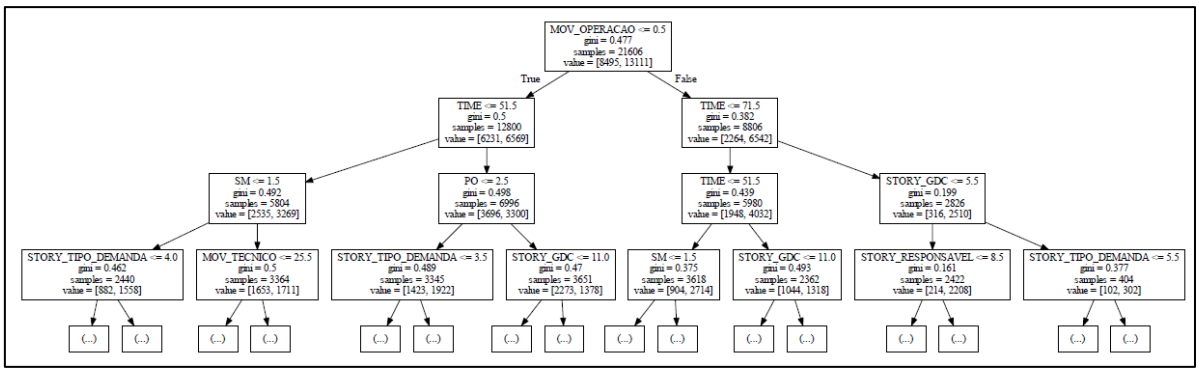
Gráfico 9 - Importância das *features* no modelo de *Machine Learning* do *dataset* removendo as operações de validação e geração



Fonte: elaborado pelo autor

Já na análise da árvore de decisão, apresentada na Figura 16, a informação do time também se destacou, mostrando que em alguns times o processo e/ou as pessoas estão mais alinhados.

Figura 16 - Árvore de decisão gerada pelo modelo de *Machine Learning* removendo as operações de validação e geração



Fonte: elaborado pelo autor

3.6 ANÁLISE INDIVIDUALIZADA POR TIME

Como em algumas análises o time apareceu como uma informação relevante, considerou-se que seria importante ter esse detalhamento individual tendo como base o modelo final gerado com todos os times. Então foram realizados todos os passos executados para o modelo geral e foi utilizado o mesmo *dataset,* sendo separado para cada time.

Comparando os resultados dos modelos gerados, para os times #1, #2 e #3 a precisão ficou igual ou muito parecida com o modelo geral, mostrando que as características e peculiaridades do time não foram impactantes. As árvores de decisão geradas também não tiveram mudanças nos principais dados de relevância para decisão. Para o time #5, como mostra a Figura 17, a precisão do modelo ficou bastante inferior ao modelo geral, o que se deve ao fato de o time considerar o seu pronto após a operação de validação, fazendo com que tenha menos casos que não concluiu dentro da *Sprint*.

Figura 17 - Índices do modelo de *Machine Learning* do *dataset* completo para o time #5

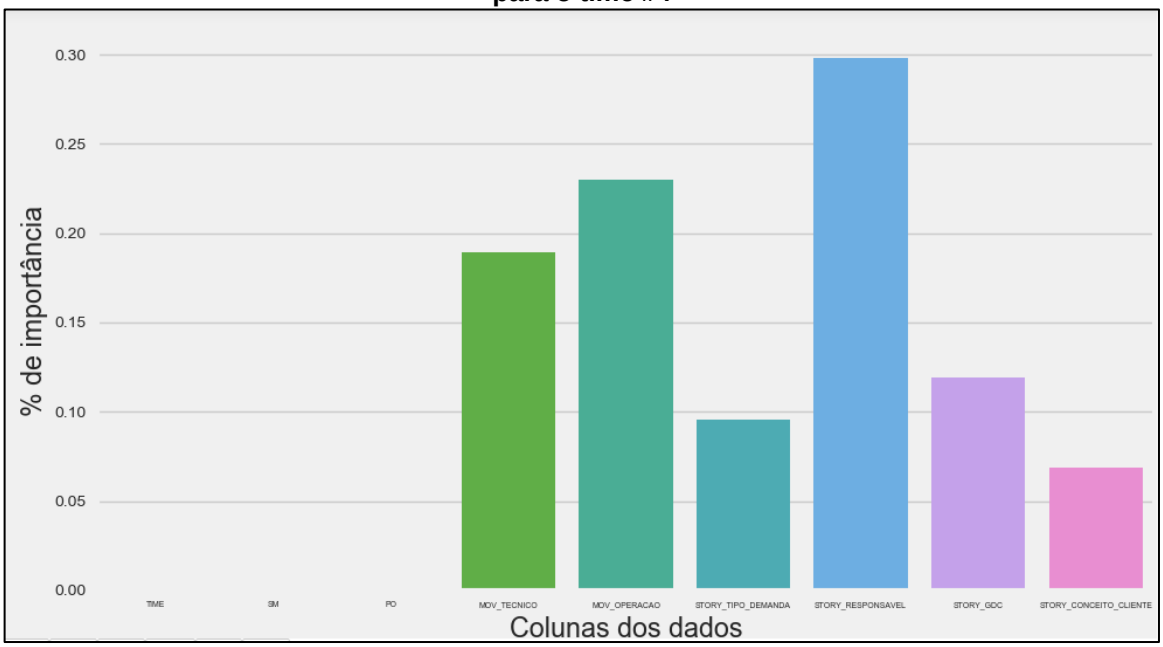
```
In [40]: from sklearn import metrics
         print(metrics.classification_report(y_teste,resultado))
```

	precision	recall	f1-score	support
0	0.56	0.30	0.39	362
1	0.96	0.99	0.97	5940
accuracy			0.95	6302
macro avg	0.76	0.64	0.68	6302
weighted avg	0.94	0.95	0.94	6302

Fonte: elaborado pelo autor

Já o time #4 apresentou uma precisão semelhante ao modelo geral, porém foi o único que apresentou na árvore de decisão e nas *features* mais relevantes à coluna do responsável da história, superior inclusive à coluna de operação, como mostra o Gráfico 10. Um dos motivos que podem explicar esse cenário é o fato de esse ser o time com maior número de demandas de infraestrutura, logo as pessoas responsáveis das histórias acabam sendo pessoas de outros times em muitos casos.

Gráfico 10 - Importância das *features* no modelo de *Machine Learning* do *dataset* completo para o time #4



Fonte: elaborado pelo autor

Também foram realizados os passos para análise de cada time considerando somente as operações de “validação” e “geração” do *dataset*. Essa análise seguiu a

mesma linha da análise geral realizada, porém quebrando os dados também por time para verificar os diferentes cenários.

De um modo geral, para os times #1, #2, #3 e #4 a precisão dos modelos ficou mais equilibrada e aumentou um pouco considerando o modelo geral, principalmente para identificar quando não conclui dentro da *Sprint.* A Figura 18 mostra o exemplo do time #3, que ficou com os melhores resultados.

Figura 18 - Índices do modelo de *Machine Learning* do *dataset* removendo as operações de validação e geração para o time #3

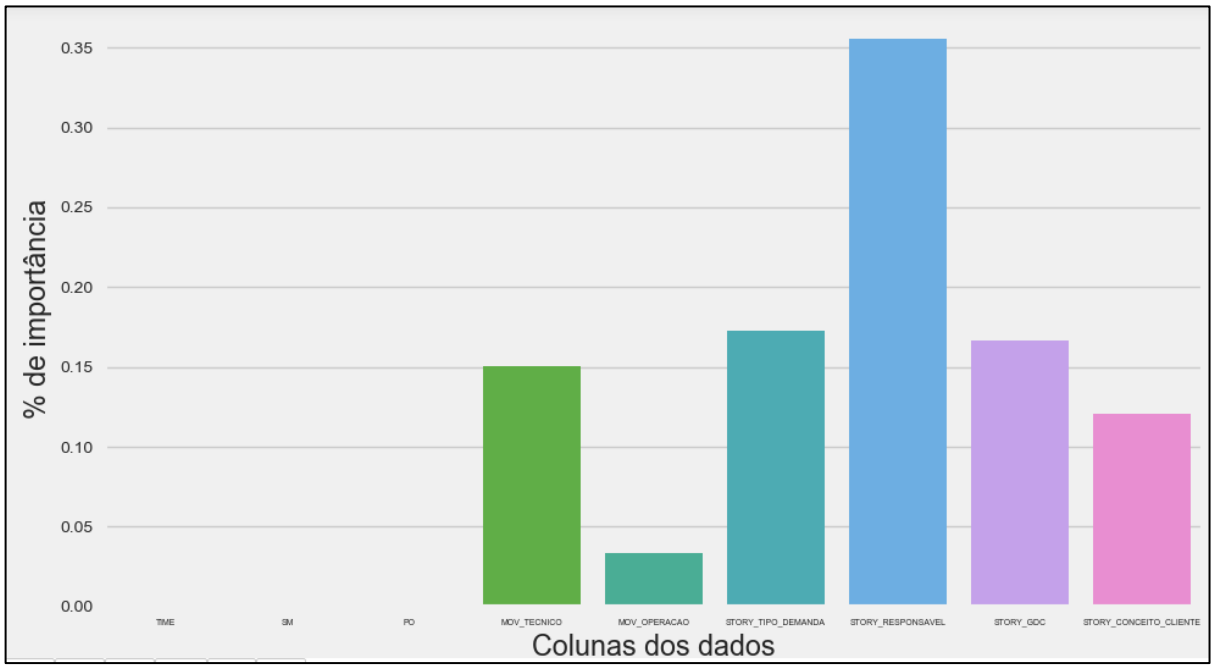
```
In [40]: from sklearn import metrics
         print(metrics.classification_report(y_teste,resultado))
```

	precision	recall	f1-score	support
0	0.76	0.83	0.79	1113
1	0.76	0.67	0.71	898
accuracy			0.76	2011
macro avg	0.76	0.75	0.75	2011
weighted avg	0.76	0.76	0.76	2011

Fonte: elaborado pelo autor

Com relação às *features* mais relevantes e à árvore de decisão gerada, todos eles passaram a ter a coluna do responsável da história como principal, como mostra o Gráfico 11 com o exemplo do time #1.

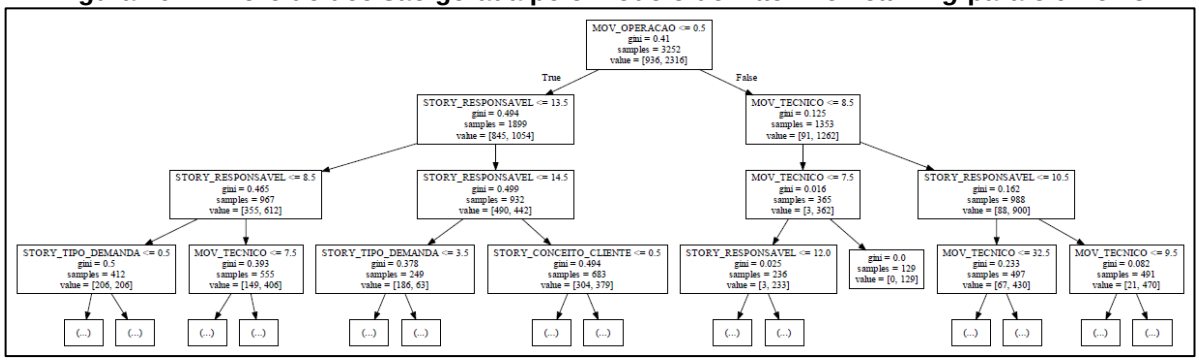
Gráfico 11 - Importância das *features* no modelo de *Machine Learning* do *dataset* removendo as operações de validação e geração para o time #1



Fonte: elaborado pelo autor

Para o time #5 a precisão ficou pior, assim como na análise anterior para o mesmo time. Para ele a coluna de operação também continuou como mais relevante, reforçando ser uma característica própria pois possui a operação de “validação” dentro de sua Definição de Pronto, como mostra a árvore de decisão gerada na Figura 19.

Figura 19 - Árvore de decisão gerada pelo modelo de *Machine Learning* para o time #5



Fonte: elaborado pelo autor

3.7 CONSIDERAÇÕES DO CAPÍTULO

Exposto o estudo realizado, não foi possível identificar, com esses dados, que há de fato problemas no processo da empresa. Os modelos gerados indicam que o problema está mais ligado às pessoas envolvidas, o que foi possível observar principalmente nas análises individuais por time e que apresentam os responsáveis das histórias como influenciadores diretos na validação das mesmas e, com isso,

também na conclusão das demandas dentro da *Sprint*. Alguns *insights* percebidos durante o estudo de caso que influenciam na conclusão das demandas dentro das Sprints foram:

- Pessoas envolvidas no processo, como responsáveis das histórias e técnico que efetua o apontamento;
- Características do time, como o tipo de demanda atendida e quantidade de demandas não planejadas na *Sprint;*
- Tipo de operação do fluxo da história, que, apesar de serem etapas do processo da empresa, indicam a atuação das pessoas envolvidas.

O capítulo a seguir revela como foram avaliados os resultados do estudo de caso e distribuídos os questionários de pesquisa, mostrando também seus resultados, por meio das respostas obtidas pelo público averiguado.

4 AVALIAÇÃO DOS RESULTADOS

Realizada a análise dos dados, estando essa devidamente fundamentada, faz-se necessário avaliar a validade dela, sustentando os resultados conforme apresentado no capítulo 3.

Dessa forma, considerou-se a elaboração de uma avaliação para verificar se os resultados condizem com os *insights* obtidos durante a análise, ou seja, mostram os mesmos comportamentos das árvores de decisão. Para tal, foram utilizados dados do período de 20/01/2020 até 20/03/2020, que é posterior ao utilizado para o estudo de caso, possuindo dados de dois meses e de 3 a 4 *Sprints* completos dos times, além de alguns dias a mais após conclusão da última *Sprint* para pegar o que não foi concluído no prazo.

Complementar a essa avaliação, efetuou-se a elaboração e posterior aplicação de um questionário com os times envolvidos no estudo. Durante o capítulo são apresentadas as respostas obtidas nele, bem como as avaliações efetuadas.

4.1 AVALIAÇÃO GERAL DOS TIMES

Para a realização da avaliação geral de todos os times não foram selecionadas as operações de “acompanhamento”, “levantamento de requisitos” e “análise”, uma vez que elas não foram consideradas relevantes durante a análise, devido ao baixo índice de apontamentos que não foram concluídos dentro da *Sprint*.

Dentre as operações consideradas, a Tabela 3 apresenta a relação de apontamentos que foram e que não foram concluídos dentro da *Sprint*, no período considerado para avaliação.

Tabela 3 - Estatística de apontamento com conclusão dentro da *Sprint* por operação utilizados na avaliação dos resultados

Operação	Concluiu na *Sprint* = Sim	Concluiu na *Sprint* = Não
Programação	99%	1%
Revisão	99%	1%
Validação	81%	19%
Geração	40%	60%

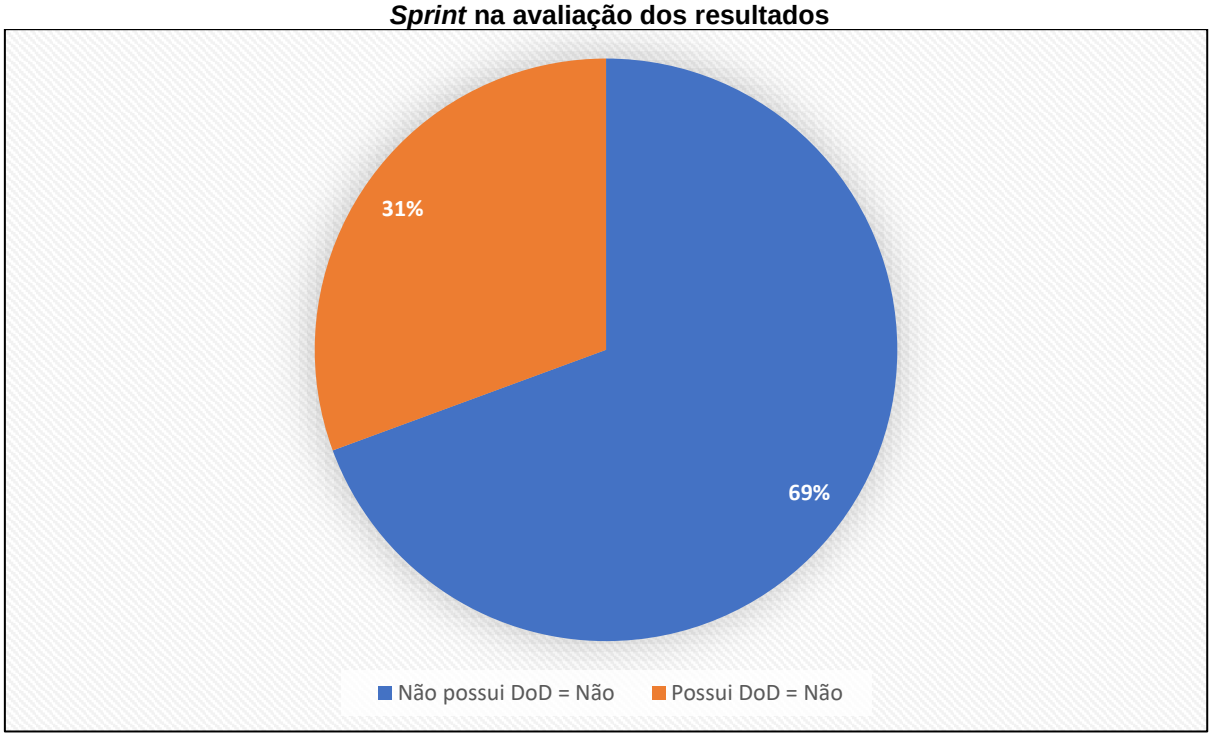
Fonte: elaborado pelo autor

É possível observar que, no período avaliado, houve mais conclusões dentro da *Sprint* nas operações de programação e revisão, que são competências do time, da mesma forma que na operação de validação, que não está nas mãos do time, com relação ao período analisado no estudo. Porém a entrega para os clientes finais,

executada imediatamente após a operação de geração, foi realizada menos vezes dentro das *Sprints*.

Isso valida a análise efetuada, mostrando que o problema está concentrado de forma mais forte na validação das histórias e acaba refletindo na geração delas devido ao processo da empresa, que possui o processo de geração semanal onde agrupa as gerações preferencialmente em um dia determinado. Aprofundando nos responsáveis das histórias envolvidos na operação de validação, foram identificadas 49 pessoas diferentes. O Gráfico 12 mostra a relação dos que tem algum apontamento que não concluiu dentro da *Sprint*, dentre esses responsáveis.

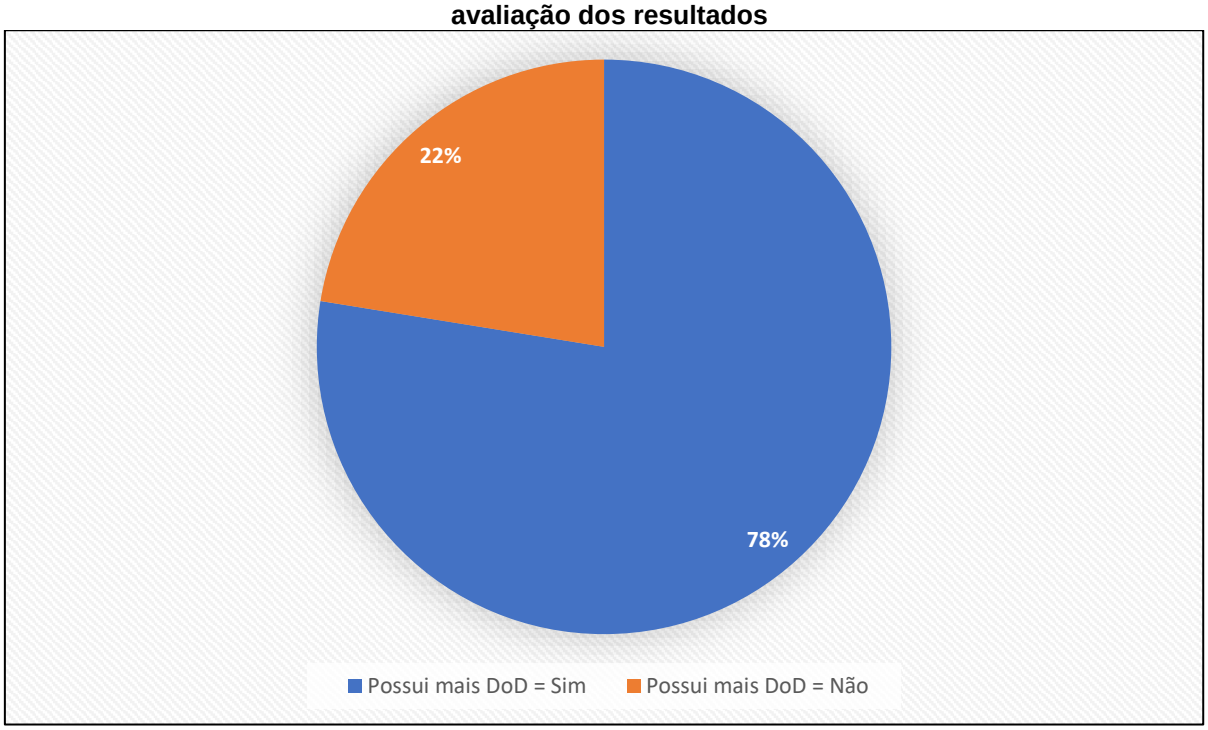
Gráfico 12 - Relação dos responsáveis que possuem apontamentos concluídos dentro da *Sprint* na avaliação dos resultados



Fonte: elaborado pelo autor

Dentre esses 31% dos responsáveis, a quantidade de apontamentos que não foram concluídos dentro da *Sprint* chega próximo a 50% dos casos. O Gráfico 13 mostra a relação dos responsáveis que possuem mais apontamentos que não concluíram dentro da *Sprint* do que concluíram dentro da *Sprint*, ou seja, erram mais que acertam se tratando da entrega das histórias.

Gráfico 13 - Relação dos apontamentos concluídos dentro da *Sprint* por responsável na avaliação dos resultados



Fonte: elaborado pelo autor

É possível identificar que o problema se concentra principalmente em 22% dos responsáveis das histórias que possuem mais casos onde não concluem dentro da *Sprint* do que concluem, fazendo com que a operação de validação atrase a entrega das histórias para os clientes finais.

4.2 AVALIAÇÃO INDIVIDUALIZADA POR TIME

Para a realização da avaliação individual entre os times também não foram consideradas as operações de “acompanhamento”, “levantamento de requisitos” e “análise”, devido ao baixo índice de apontamentos que não foram concluídos dentro da *Sprint*.

Outra situação que aconteceu foi o fato de o time #2 ter sido juntado ao time #3 um pouco antes do período de avaliação estipulado. Devido a isto, a avaliação foi realizada individualmente somente para os times #1, #4 e #5, para não distorcer os dados.

Dentre as operações consideradas, a Tabela 4 apresenta a relação de apontamentos que foram concluídos dentro da *Sprint* e que não foram, no período considerado para avaliação, para os times avaliados.

Tabela 4 - Estatística de apontamento com conclusão dentro da *Sprint* por operação utilizados na avaliação dos resultados por time

Time	Operação	Concluiu na *Sprint* = Sim	Concluiu na *Sprint* = Não
#1	Programação	98%	2%
	Revisão	98%	2%
	Validação	53%	47%
	Geração	33%	67%
#4	Programação	99%	1%
	Revisão	100%	0%
	Validação	93%	7%
	Geração	49%	51%
#5	Programação	100%	0%
	Revisão	100%	0%
	Validação	100%	0%
	Geração	49%	51%

Fonte: elaborado pelo autor

Esse resultado também valida o estudo realizado, porque mostra que as diferentes características dos times influenciam na entrega final. Os times #4 e #5 tiveram resultados melhores com relação a operação de validação, sendo que ambos possuem como característica terem mais demandas de projetos e infraestrutura e o time #5 possui a validação dentro de sua Definição de Pronto. Ao mesmo tempo a operação de geração não foi muito afetada, visto que algumas demandas maiores e de projetos são geradas após mais *Sprints*, realizando a entrega da alteração completa.

4.3 ANÁLISE DO QUESTIONÁRIO APLICADO

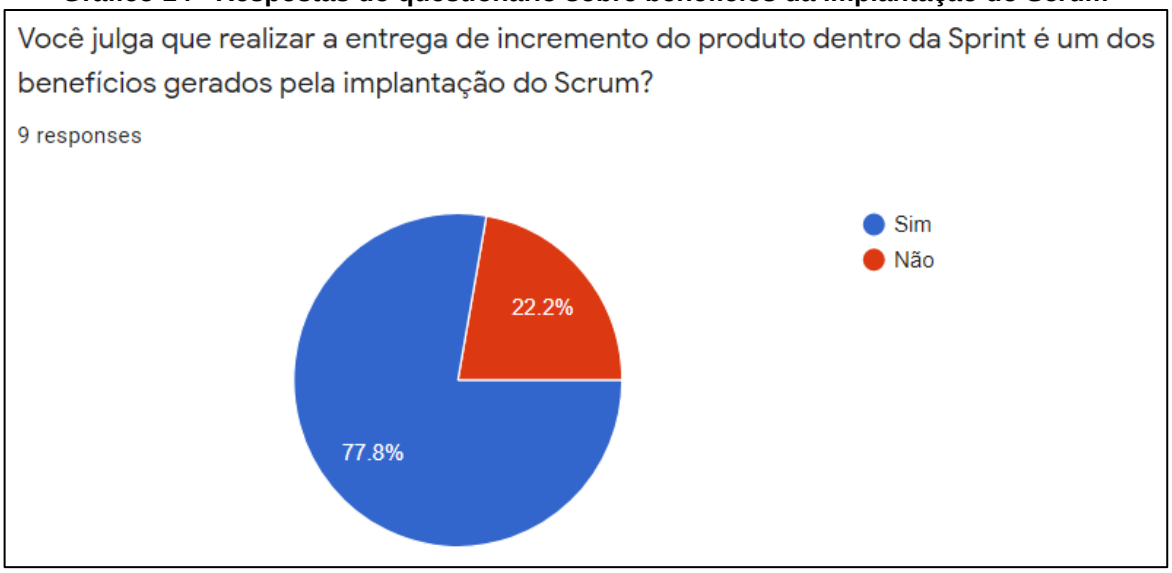
Da mesma forma que a avaliação individual por time, o questionário foi aplicado somente nos times #1, #4 e #5. Ao todo foram apresentadas 10 questões, que estão disponíveis no Apêndice A deste trabalho, para 14 integrantes dos três times mencionados, sendo obtidas 9 respostas. As questões foram embasadas a partir do referencial teórico em conjunto com os *insigths* obtidos durante a realização da análise dos dados do estudo de caso.

O questionário teve alguns objetivos definidos inicialmente, buscando identificar a percepção do time e seu nível de satisfação com relação à Definição de Pronto que cada um possui. Também procura a opinião dos times sobre a atuação das pessoas, em especial os responsáveis das histórias, no processo, fazendo ligação com a análise efetuada no capítulo 3 que apresentou esse problema.

Seguindo o que diz Cohn (2011), espera-se que os Times *Scrum* entreguem algo de valor para os clientes a cada *Sprint*. E de um modo geral os times concordam que

a entrega de incrementos do produto dentro da *Sprint* é um dos benefícios gerados pela implantação do *Scrum*, como mostra o Gráfico 14.

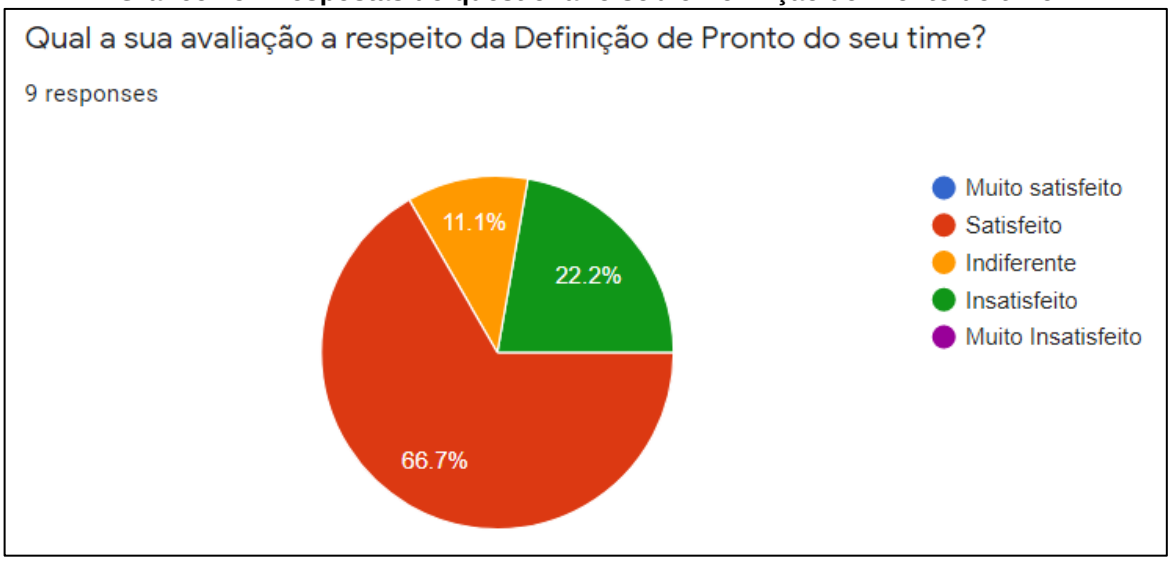
Gráfico 14 - Respostas do questionário sobre benefícios da implantação do *Scrum*



Fonte: elaborado pelo autor

Porém quando questionados sobre a Definição de Pronto do seu time, a maioria indicou estar satisfeito, conforme mostra o Gráfico 15. Isso acaba sendo um reflexo da aceitação dos integrantes com relação ao processo no qual está inserido, porque nas respostas foi mencionado que o time trabalha até onde tem alcance e que há um isolamento do time com as outras partes da empresa, além da indicação que a Definição de Pronto atual atende à necessidade do time.

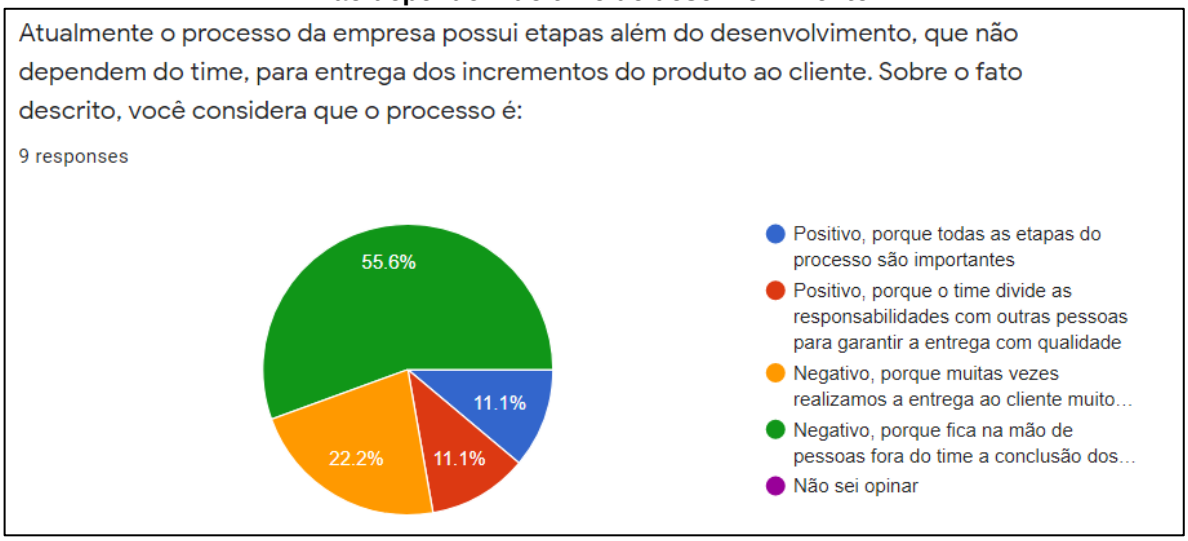
Gráfico 15 - Respostas do questionário sobre Definição de Pronto do time



Fonte: elaborado pelo autor

O estudo de Quadros (2017) indicou que ao aplicar *Scrum* em uma empresa ERP muitas *Sprints* não eram concluídas por terem etapas que não dependiam do time desenvolvimento, o que é a mesma situação da empresa estudada neste trabalho. Ao serem questionados sobre este fato, de um modo geral os times consideram como negativo, porque foge do seu controle e fica na mão de outras pessoas a conclusão das demandas para ser possível realizar sua entrega, como mostra o Gráfico 16.

Gráfico 16 - Respostas do questionário sobre o processo da empresa que possui etapas que não dependem do time de desenvolvimento

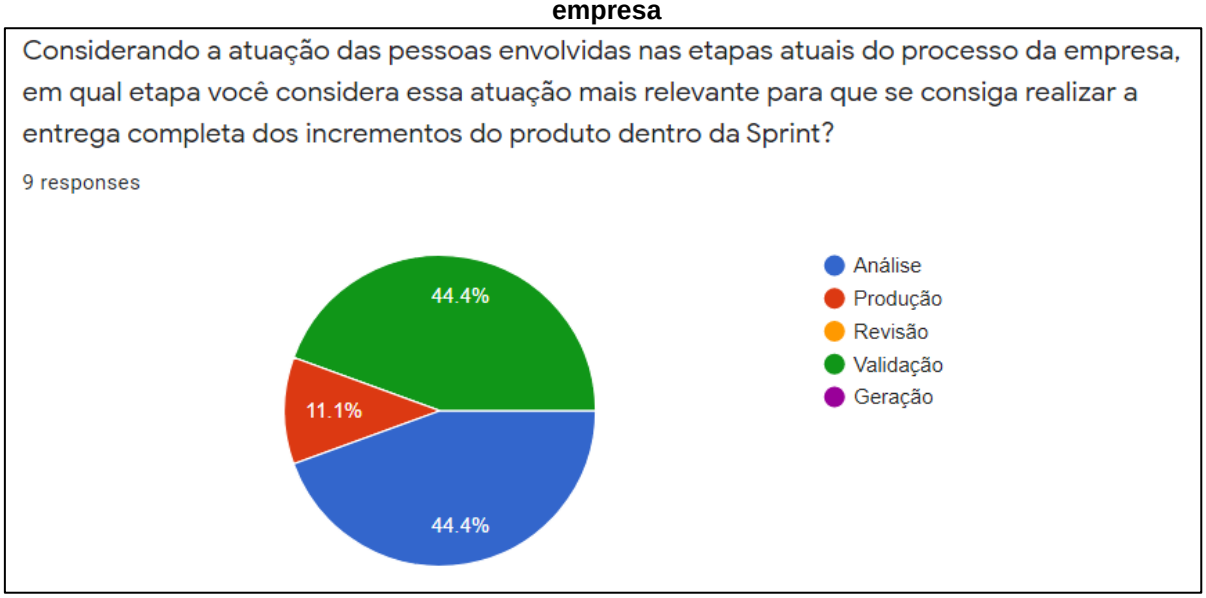


Fonte: elaborado pelo autor

Já Araújo e Dielle (2015) também fizeram um estudo onde foi implantado *Scrum* em uma empresa ERP e havia entregas após a conclusão da *Sprint*, porém os demais benefícios adquiridos fizeram com que isso não fosse tão relevante. Algo similar pode ser observado nas respostas dos times, uma vez que alguns integrantes consideram positivo o fato de mais pessoas serem envolvidas, devido a importância das etapas e para garantir uma maior qualidade.

Com relação às operações analisadas no estudo de caso, onde a validação se mostrou a principal influência para a conclusão ou não das histórias dentro da *Sprint*, de um modo geral ela foi mencionada também pelos times como mais relevante, assim como a análise, como mostra o Gráfico 17.

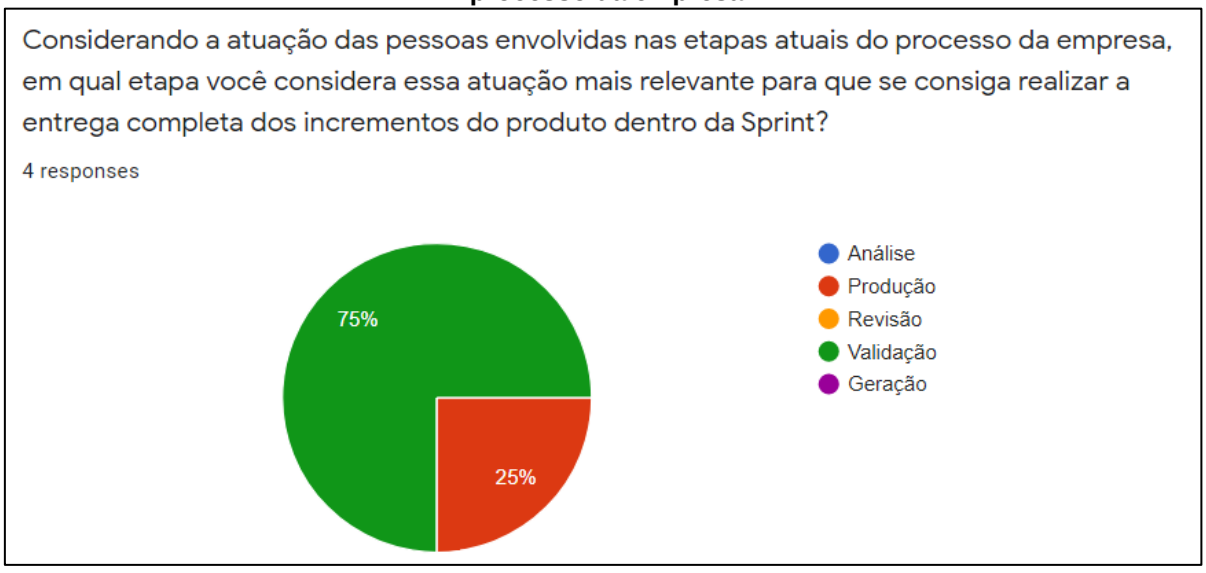
Gráfico 17 - Respostas do questionário sobre etapa mais relevante para o processo da empresa



Fonte: elaborado pelo autor

O fato de a análise ser mencionado como operação mais relevante por boa parte dos integrantes mostra uma percepção diferente deles com relação ao que o estudo mostrou que de fato ocorre. Talvez eles procurem alguma razão quando o time não consegue concluir alguma demanda e identificam que os imprevistos gerados pela análise mal feita de uma história seja muito relevante, o que não foi indicado pelo estudo, que mostrou que é muito baixo o índice de casos não concluídos devido a problemas no time desenvolvimento de fato, através das operações de análise, produção ou revisão.

Gráfico 18 - Respostas do time #1 para o questionário sobre etapa mais relevante para o processo da empresa



Fonte: elaborado pelo autor

Já o time #1 possui uma percepção mais próxima ao que mostrou o estudo de caso, como mostra o Gráfico 18, o que acaba reforçando os resultados obtidos. Esse time possui mais casos em que não conclui a história dentro da *Sprint*, como mostrou a avaliação feita para validar a análise, e com um maior índice devido a validação, que fica refletido na opinião do time.

4.4 CONSIDERAÇÕES DO CAPÍTULO

A avaliação valida os resultados obtidos no estudo de caso, reforçando que a entrega de incrementos do produto dentro das *Sprints* está mais ligado às pessoas, considerando o processo da empresa desenvolvedora de ERP. As características dos times analisados deixam o resultado mais evidente, indicando que o nível de maturidade deles pode fazer com que evoluam na sua Definição de Pronto, porém precisam estar alinhados com as demais pessoas envolvidas para que seja efetiva.

Já os questionários aplicados indicam que os times entendem que no *Scrum* o incremento do produto deve ser entregue dentro da *Sprint*, porém de um modo geral estão satisfeitos com a Definição de Pronto que possuem. Também julgam como negativo o fato de a empresa possuir em seu processo etapas que não dependem do time desenvolvimento para realizar a entrega dos incrementos aos clientes, indicando principalmente a etapa de validação como relevante para conclusão das demandas, considerando a atuação das pessoas envolvidas.

Com os dados analisados, não foi possível observar problemas com relação à Definição de Pronto dos times, visto que alguns times obtiveram resultados parecidos com definições diferentes. A aplicação do questionário mostrou que não é um problema do ponto de vista de entrega, mas sim de gestão das pessoas que estão incorporadas no processo, uma vez que elas relatam desconforto com o que acontece.

É importante também ressaltar que os resultados obtidos podem ser considerados por outras empresas que desenvolvem *software*, uma vez que apresentou cenários semelhantes aos encontrados nos trabalhos relacionados, como as demandas de erros e legislações que existem. Porém as peculiaridades de cada empresa são muito relevantes, pois indicam como o processo absorve as características do sistema, em especial o ERP, e como se dá o envolvimento das pessoas.

5 CONSIDERAÇÕES FINAIS

As organizações têm se tornado mais ágeis, sendo o *Scrum* o *framework* mais utilizado entre os métodos ágeis conhecidos no mercado, buscando resultar em uma maior eficiência em seus processos. Na teoria, as entregas de incrementos do produto devem acontecer com frequência, onde ao final de cada *Sprint* o time de desenvolvimento entrega um incremento de funcionalidade que deve estar na condição de ser utilizado e atender a sua Definição de Pronto.

Na indústria de *softwares,* as empresas não têm um histórico de realizar a coleta de métricas e explicar o que foi obtido, apesar dos benefícios, porque é trabalhoso e nem sempre se tem conclusões fáceis (COHN, 2011). A partir disso, o presente trabalho propôs realizar uma análise na perspectiva de dados obtidos em um caso real. Iniciando numa linha de estatística, viu-se que seria muito rasa para o tipo de análise desejado e partiu-se para a Mineração dos Dados, realizando uma análise preditiva e gerando previsões e *insights* a partir dos cenários expostos de desenvolvimento e entrega.

Um dos problemas motivadores da pesquisa trata a demora para conclusão de todas as etapas do processo organizacional, onde algumas dessas são realizadas somente após a conclusão das *Sprints,* gerando questionamentos com relação à problemas no processo. Em um dos times participantes do estudo de caso, por exemplo, a etapa de validação das histórias desenvolvidas, que não compete ao time de desenvolvimento, foi mencionada como mais relevante para que se consiga realizar a entrega dos incrementos do produto. Este é um indício do problema que havia sido identificado ao início da pesquisa.

Com isso, explorou-se na teoria alguns conceitos relacionados a entregas em ambiente ágil e Definição de Pronto em Times *Scrum*. Também foram buscadas características e peculiaridades de sistemas ERP, procurando identificar, através de trabalhos relacionados, o que já foi estudado sobre o tema proposto.

Conforme detalhado no capítulo 3, em conjunto com um especialista, que possui atuação acadêmica relevante na área de Mineração de Dados, foi preparado um *dataset* a partir dos apontamentos internos disponíveis no banco de dados da empresa estudada e aplicada técnica de mineração para gerar previsões e identificar características do modelo montado. Com isso, foi possível gerar *insights* do que afeta

a Definição de Pronto dos times e observar quando as demandas não são concluídas dentro das *Sprints.*

Como contribuição de pesquisa no presente trabalho, pode ser destacado que o estudo de caso procurou evidenciar possíveis problemas no processo de geração e entrega contínua em ambiente de desenvolvimento de *software* com *Scrum*, que é um tema importante de ser estudado porque é um dos principais benefícios que se busca com a implantação do *Scrum.* Porém, entende-se que a análise com dados é crítica e fazer um diagnóstico do cenário é muito importante para evitar que sejam feitas mudanças a partir do achismo e que não trariam os ganhos esperados.

Como trabalho futuro, existe o interesse em explorar outros dados disponíveis que sejam relevantes do ponto de vista de entrega, para reforçar que os problemas não estão no processo, mas sim na gestão das pessoas envolvidas e verificar se os mesmos resultados podem ser considerados por outras empresas desenvolvedoras de *software*. Também pode ser explorada a maturidade dos times para verificar o quanto isso afeta nas entregas realizadas pelos mesmos.

A partir do que foi apresentado, conclui-se que o trabalho atingiu seu objetivo geral e alcançou seus objetivos específicos, apresentando contribuições de conhecimento científico através do processo de análise realizado. Portanto, é uma experiência válida, fortalecendo os conhecimentos adquiridos com a prática com os Times *Scrum*.

REFERÊNCIAS BIBLIOGRÁFICAS

ARAÚJO, Marco Antônio P.; DIELLE, Daniel F. 2015. **Desafios na Implementação do Scrum: um Estudo de Caso Sobre a Utilização da Metodologia Ágil em uma Empresa Desenvolvedora de Software.** Disponível em: <https://seer.cesjf.br/index.php/cesi/article/view/516/401>. Acesso em: out. 2019.

AUDY, Jorge. **Scrum 360**: Um guia completo e prático de agilidade. São Paulo, SP: Casa do Código, 2015.

BECK, Kent et al. **Manifesto Ágil**. 2001. Disponível em: <http://www.manifestoagil.com.br/>. Acesso em: out. 2019.

CEPEDA, Claudia E.; COUTINHO, Maria Lúcia G.; VIGNA, Claudio M. 2017. **Causa do atraso de projetos: Análise das causas mais relevantes para o atraso de projetos de software.** Disponível em: <https://doi.org/10.5585/iptec.v6i2.150>. Acesso em: out. 2019.

CHOPRA, S.; MEINDL, P. **Gerenciamento da Cadeia de Suprimentos**: Estratégia, Planejamento e Operação. São Paulo: Prentice Hall, 2003.

COHN, Mike. **Desenvolvimento de software com Scrum**: Aplicando métodos ágeis com sucesso. Porto Alegre, RS: Bookman, 2011.

CUNHA, Thiago Ferraz V. da; ANDRADE, Rossana M. C. Agile DMAIC: **Um Método para Avaliar e Melhorar o Uso do Scrum em Projetos de Software**. In: SIMPÓSIO BRASILEIRO DE QUALIDADE DE SOFTWARE, 13., 2014, Blumenau. Anais... Blumenau: FURB, 2014. p. 121-135.

GOLDSCHMIDT, Ronaldo; PASSOS, Emmanuel. **Data Mining**: Um Guia Prático. Rio de Janeiro, RJ: Elsevier, 2005.

GOMES, André Faria. **Agile**: Desenvolvimento de software com entregas frequentes e foco no valor de negócio. São Paulo, SP: Casa do Código, 2014.

GRUS, Joel. **Data Science do Zero**: Primeiras Regras com o Python. Rio de Janeiro, RJ: Alta Books, 2016.

MASSARI, Vitor. **Agile Scrum Master no gerenciamento avançado de projetos**. Rio de Janeiro, RJ: Brasport, 2016.

PECHARROMÁN, Aitor Urteaga. 2015. **Aplicación de la metodología de desarrollo ágil Scrum para el desarrollo de un sistema de gestión de empresas.** Disponível em: <http://hdl.handle.net/10016/23750>. Acessado em: out. 2019.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de software**: uma abordagem profissional. 8. ed. Porto Alegre, RS: AMGH, 2016.

PRIKLADNICKI, Rafael; WILLI, Renato; MILANI, Fabiano (Org.). **Métodos ágeis para desenvolvimento de software**. Porto Alegre, RS: Bookman, 2014.

PRODANOV, Cleber Cristiano; FREITAS, Ernani Cesar de. **Metodologia do trabalho científico: métodos e técnicas da pesquisa e do trabalho acadêmico**. 2. ed. Novo Hamburgo, RS: Feevale, 2013.

QUADROS, Demitrius Ruan. 2017. **Aplicando metodologias ágeis em projetos de integração de sistemas.** Disponível em: <https://publicacao.uniasselvi.com.br/index.php/TI_EaD/article/download/1688/802>. Acesso em: out. 2019.

RODRIGUES, Vitor. 2019. **Métricas de Avaliação: acurácia, precisão, recall… quais as diferenças?** Disponível em: <https://medium.com/@vitorborbarodrigues/m%C3%A9tricas-de-avalia%C3%A7%C3%A3o-acur%C3%A1cia-precis%C3%A3o-recall-quais-as-diferen%C3%A7as-c8f05e0a513c/>. Acesso em: mar. 2020.

RUBIN, Kenneth S. Essential Scrum: **A practical guide to the most popular agile process**. Estados Unidos: Addison-Wesley, 2012.

SABBAGH, Rafael. **Scrum: Gestão ágil para projetos de sucesso**. São Paulo, SP: Casa do Código, 2014.

SANTANA, Felipe. 2017. **Afinal, o que é Big Data e Mineração de dados?** Disponível em: <https://minerandodados.com.br/oque-big-data-mineracao-de-dados/>. Acesso em: mar. 2020.

SANTANA, Felipe. 2019. **O que é Data Science?** Disponível em: <https://minerandodados.com.br/o-que-e-data-science/>. Acesso em: mar. 2020.

SANTANA, Felipe. 2020. **Cientista de Dados – Curiosidades e Desafios da Profissão.** Disponível em: <https://minerandodados.com.br/data-scientist-curiosidades-e-desafios-da-profissao/>. Acesso em: mar. 2020a.

SANTANA, Felipe. 2020. **Árvores de Decisão.** Disponível em: <https://minerandodados.com.br/arvores-de-decisao-conceitos-e-aplicacoes/>. Acesso em: mar. 2020b.

SANTANA, Rodrigo. 2018. **Matriz de Confusão, Você Sabe Utilizar?** Disponível em: <https://minerandodados.com.br/matriz-de-confusao/>. Acesso em: mar. 2020.

SCHWABER, Ken; SUTHERLAND, Jeff. **The Scrum Guide: The definitive Guide to Scrum: The rules of the game**. 2017. Disponível em: <https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf>. Acesso em: ago. 2019.

SILVA, Leandro Augusto da; PERES, Sarajane Marques; BOSCARIOLI, Clodis. **Introdução à Mineração de Dados**: Com Aplicações em R. Rio de Janeiro, RJ: Elsevier, 2016.

SILVA, Marcos Antônio da; OLIVEIRA, Paloma Maira. 2016. **Desenvolvimento de sistemas integrados de gestão empresarial para micro e pequenas empresas.** Disponível em: < https://pmkb.com.br/artigos/desenvolvimento-de-sistemas-integrado-de-gestao-empresarial-para-micro-e-pequenas-empresas-utilizando-metodologias-de-processos-ageis/>. Acesso em: out. 2019.

SOUZA, J., SILVA, D. 2015. **Cultura Organizacional: O Fator Chave para o Sucesso da Implantação de um Sistema ERP**. Disponível em: <http://177.107.89.34:8080/jspui/bitstream/123456789/313/1/SouzaSilva.pdf>. Acesso em: set. 2019.

SUTHERLAND, Jeff. **Scrum**: a arte de fazer o dobro do trabalho na metade do tempo. São Paulo, SP: LeYa, 2014.

VERSION ONE. **The 13th Annual State of Agile Report**. 2019. Disponível em: <https://explore.versionone.com/state-of-agile/13th-annual-state-of-agile-report>. Acesso em: ago. 2019.

APÊNDICE A – QUESTIONÁRIO PARA AVALIAÇÃO DO ESTUDO

A seguir, apresenta-se o questionário respondido por 9 desenvolvedores que fazem parte dos times estudados. As respostas obtidas foram apresentadas em forma de resultados no capítulo 4 do presente trabalho.

Estudo de Caso sobre Definição de Pronto para Times Scrum que desenvolvem ERP

Este instrumento tem como objetivo coletar a percepção dos times envolvidos na pesquisa "Definição de Pronto para Times Scrum que Desenvolvem ERP: Um Estudo de Caso em Ambiente Ágil Baseado em Dados", a qual, fundamentada nos princípios do Manifesto Ágil, propõe a elaboração de uma metodologia de suporte à definição de pronto a partir de uma análise qualitativa no fluxo de entrega dos incrementos do produto, para times Scrum que desenvolvem ERP.

Este questionário não exige identificação nominal, garantindo a privacidade das informações fornecidas por você. Os dados solicitados serão utilizados unicamente para caracterizar os participantes do estudo. Não são conhecidos riscos aos respondentes e sua participação é totalmente voluntária.

Desde já agradecemos sua colaboração!

Autores: Thiago Weber (thiagoweber00@hotmail.com) e Prof. Dra. Adriana Neves dos Reis.

* Required

Qual a sua avaliação a respeito da Definição de Pronto do seu time? *

- ◯ Muito satisfeito
- ◯ Satisfeito
- ◯ Indiferente
- ◯ Insatisfeito
- ◯ Muito Insatisfeito

Por que? *

Your answer

Você julga que realizar a entrega de incremento do produto dentro da Sprint é um dos benefícios gerados pela implantação do Scrum? *

- ◯ Sim
- ◯ Não

Atualmente o processo da empresa possui etapas além do desenvolvimento, que não dependem do time, para entrega dos incrementos do produto ao cliente. Sobre o fato descrito, você considera que o processo é: *

- ◯ Positivo, porque todas as etapas do processo são importantes
- ◯ Positivo, porque o time divide as responsabilidades com outras pessoas para garantir a entrega com qualidade
- ◯ Negativo, porque muitas vezes realizamos a entrega ao cliente muito após o término da Sprint
- ◯ Negativo, porque fica na mão de pessoas fora do time a conclusão dos incrementos do produto
- ◯ Não sei opinar

Com relação aos responsáveis das histórias realizarem a validação dela após conclusão por parte do time desenvolvimento, você considera: *

- [ ] Importante para garantir que atende a necessidade do cliente
- [ ] Ruim a falta de aproximação entre o time desenvolvimento e os responsáveis das histórias
- [ ] Os responsáveis das histórias não veem tanta importância na entrega do incremento do produto dentro da Sprint
- [ ] Os responsáveis das histórias não colocam como prioritária essa validação dentre suas atividades diárias
- [ ] Não sei opinar

Considerando a atuação das pessoas envolvidas nas etapas atuais do processo da empresa, em qual etapa você considera essa atuação mais relevante para que se consiga realizar a entrega completa dos incrementos do produto dentro da Sprint? *

- ◯ Análise
- ◯ Produção
- ◯ Revisão
- ◯ Validação
- ◯ Geração

A Definição de Pronto do time deveria contemplar a entrega de incremento do produto ao cliente final dentro da Sprint, considerando o processo da empresa? *

Em um nível de 0 a 10, sendo 0 discordo totalmente e 10 concordo totalmente:

1	2	3	4	5	6	7	8	9	10
◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

Como você se sente ao não concluir uma tarefa dentro da Sprint? *

Your answer

Como você se sente ao não realizar a entrega de uma tarefa para o cliente final dentro da Sprint? *

Your answer