Rodrigo Bevilacqua Marcondes¹; Marcos Aurélio Cordeiro¹;

Wesley de Araujo2; Felipe Fillus2

***¹*** ¹*Docente do curso de Sistemas de Informação da UNIFACEAR Centro Universitário*

*²Graduando(a) do curso de Sistemas de Informação e análise de sistemas da UNIFACEAR*

**RESUMO**

*Hoje, grande parte das empresas do cotidiano busca um mercado que evolui e amplia seus negócios sempre, e onde o custo da mão-de-obra é reduzido e as vendas com seus produtos ou negócios aumentam, mas nem sempre é evidente que os negócios são organizados de forma que sejam ágeis para uma empresa em questão. Assim, nosso objetivo conjunto com as disciplinas de Sistema de Informação III e Análise de Sistemas III da UNIFACEAR foi criar um aplicativo para o ambiente de trabalho de desktop que permita o gerenciamento de dados daqueles que pretendem se candidatar a campeonatos e se familiarizar ainda mais com o mundo da automóvel.*

*Palavras chave: kartódromos, tecnologia , veículos, aplicativos desktop.*

***ABSTRACT***

*Today, a large part of the companies in the daily, search for a market that evolves and grows their business always, and where the cost of labor is reduced, and sales with their product or business increased, but not always it is evident that business are organized so that it is agile for a company in question. Thus, our joint goal with the disciplines of Information System III and Systems Analysis III at UNIFACEAR was create an application for the desktop work environment that allows the management of data of those who intend to apply for championships and familiarize themselves further with the world of the automobile.*

*Key Words: karting tracks, technology, vehicles, desktop applications.*

**1. INTRODUÇÃO**

O kart surgiu em meados dos anos 50 nos EUA onde os após a segunda guerra mundial, ele inicialmente foi projetado por pilotos de aviões onde o foco era apenas a diversão dos mesmos, fato que hoje em dia virou algo que é conhecido como um esporte praticado mundialmente por jovens e adultos de todas as idades.

No ano 1956 foi o primeiro kartódromo, que consiste em uma pista para treinos e corridas de kart, no sul da Califórnia nos EUA por Art Ingels que é conhecido mundialmente como “o pai do kart”. No Brasil o mesmo chegou por volta dos anos 60 com uma grande corrida chamada “500 Milhas de Kart Granja Vianna”, que consistia em uma corrida de persistência no qual os pilotos tinham que passar por uma prova de 12 horas e se passava no município Cotia, localizado em São Paulo. Hoje os Kartódromos e os Karts são espalhados por todo o mundo tendo pistas de diversos tipos e tamanhos.

Esse artigo está sendo feito com o propósito de relatar do que se trata o projeto e de como ele foi feito. Para retratar e informar as suas funcionalidades e características serão utilizados certos métodos da linguagem UML, esses que descreverão a noção do que e como o projeto do sistema de gestão funcionará.

**DESENVOLVIMENTO**

Considerando o crescente mercado de kartódromos e pilotos que desejam entrar no mundo do automobilismo profissional, fora desenvolvido o aplicativo “KartOnRoad” que consiste em melhorar o relacionamento entre estes pilotos e o setor de administração do kartódromo.

O projeto do KartOnRoad consiste em desenvolver o sistema de gestão de kartódromo cujo o objetivo ser o de permitir seus usuários(Pilotos) de se cadastrarem e logarem no sistema para poder participar das corridas que desejarem. Podendo entrar em corridas normais ou campeonatos que serão realizadas ou criar as suas próprias corridas. Além de poder enviar avisos aos demais participantes da corrida ou convidar outros pilotos, caso o piloto seja o ADM da corrida.

Para documentá-lo foram usados métodos da linguagem UML, esses que servirão para registrar, explicar as funções e características do sistema de KartOnRoad. O documento será composto pelo PMCanvas, a análise de concorrência, os requisitos funcionais e não funcionais, as regras de negócio e o casos de uso, o diagrama de entidade-relacionamento, o modelo de processo de negócio e diagrama de classes para que desse jeito, o software vai ficar compreensível como um todo (VIEGAS, 2009).

* 1. PMCANVAS

Project *Model Canvas (PMCanvas)* é uma ferramenta usada para desenvolver projetos de uma maneira rápida e em conjunto, porque prevê a participação de vários profissionais que têm conhecimento e poder de decisão sobre o projeto, permitindo que todos os envolvidos nas ideias participem de forma colaborativa, de maneira que vão dar consistência ao trabalho (BUILDER, 2017).

Pode ser definida também como uma metodologia robusta, porém simples, de planejamento de projetos. Os seus componentes estão organizados em blocos de perguntas (Por quê, O quê, Quem, Como, Quando e Quanto) que estão integrados de forma que seja possível gerenciar um projeto, como mostrado na Figura 1 (BUILDER, 2017).



FIGURA 1: PMCANVAS DO PROJETO

FONTE: OS AUTORES

* 1. ANÁLISE DE CONCORRÊNCIA

A análise de concorrência por sua vez, é utilizada de forma estratégica entre empresas, tem como objetivo analisar e estudar os cenários onde softwares parecidos ou com o mesmo objetivo atuam, e a forma com que eles se relacionam com o cliente final, sendo assim servindo como um auxílio para fortalecer a posição da empresa no mercado ou até mesmo para auxiliar na criação de novas ideias para novos produtos para a empresa.

Na figura 02 está uma análise de concorrência entre KartOnRoad e RaceTracksKGV.



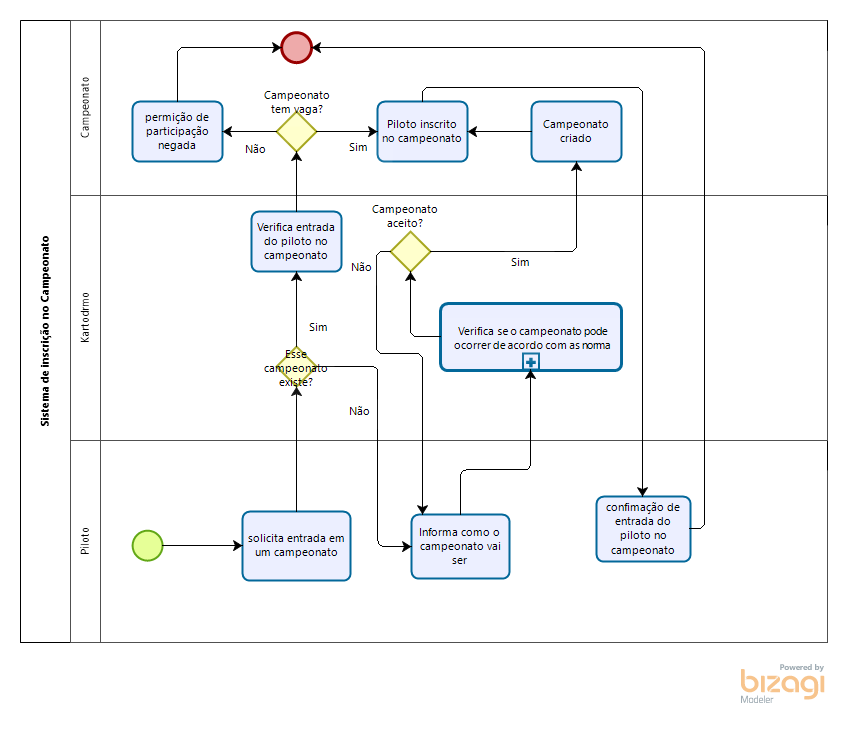
FIGURA 02: ANÁLISE DE CONCORRÊNCIA

FONTE: OS AUTORES (2020)

.

* 1. MODELO DE PROCESSO DE NEGÓCIO (BPM)

Modelagem de Processos de Negócio (*Business Process Modeling)* é uma demonstração dos processos de negócio de uma empresa ou organização, que tem como objetivo documentar e analisar os processos praticados na mesma. Utilizando BPM é possível compreender as atividades, e mostrar como deve ser executado os processos da empresa. Na figura 3 existe um diagrama de modelagem de processos de negócio mostrando como ocorre as inscrições dos pilotos nas corridas.

FIGURA 03: MODELO DE PROCESSO DE NEGÓCIO

FONTE: OS AUTORES (2020)

* 1. REQUISITOS FUNCIONAIS

Os requisitos funcionais representam as necessidades que o sistema deve prover.

As colunas possuem os seguintes significados para a tabela 1:

1. RF – Identificação do Requisito, composto pela primeira letra da característica de qualidade mais numeração sequencial.
2. Descrição – descrição do requisito.
3. Detalhamento – Detalhamento do requisito
4. Requisitos relacionados – outros requisitos que se relacionam com esse requisito.
5. Prioridade – prioridade do requisito.
6. RN – identificação de regra de negócio.

TABELA 01 – REQUISITOS FUNCIONAIS

|  |  |  |  |
| --- | --- | --- | --- |
| RF | Descrição | Prioridade | Requisitos relacionados |
| RF 1 | O sistema deve cadastrar/logar os pilotos | Alta | RN n°8,RN n°3 e RF n°7 |
| RF 2 | O sistema deve cadastrar/logar os Kartódromos. | alta | RF n°7 |
| RF 3 | O sistema deve dar permissão ao Pilotos/ADM o de criar/cancelar campeonatos. | alta | RN n°2 e RF n°12 |
| RF 4 | O sistema deve disponibilizar datas livres para corridas. | média | nenhuma |
| RF 5 | O sistema deve ter uma classificação dos campeonatos. | baixa | RN n°6 |
| RF 6 | O sistema deve permitir o piloto avaliar os kartódromos. | média | nenhuma |
| RF 7 | O sistema deve enviar email com o código de login quando solicitado. | média | nenhuma |
| RF 8 | O sistema deve disponibilizar contato com o usuário via email. | média | nenhuma |
| RF 9 | O sistema deve gerar os tempos dos pilotos nas corridas. | alta | RN n°13 |
| RF 10 | O sistema deve gerar o ganhador de um campeonato. | alta | RF n°16 e RN n°15 |
| RF 11 | O sistema deve banir qualquer piloto que levar 3 strikes. | média | RN n°8, RN n°10, RF n°8 e RN n°11 |
| RF 12 | O sistema deve permitir pilotos/ADM Convidarem outros pilotos para determinada corrida. | alta | nenhuma |
| RF 13 | O sistema deve permitir o piloto/ADM fazer a chamada dos pilotos antes de iniciar uma corrida. | média | RF n°16, RN n°1 e RN n°11 |
| RF 14 | Os pilotos/ADM podem gerar mensagens de avisos aos Pilotos que estão participando. | alta | nenhuma |
| RF 15 | O sistema deve permitir os pilotos negarem/aceitarem convites. | baixa | RF n°12 |
| RF 16 | O sistema deve permitir o piloto ADM iniciar um campeonato. | alta | RF n°13, RN n°13 e RNn°14 |
| RF 17 | O sistema deve permitir o piloto se remova sua inscrição de um campeonato. | média | nenhuma |
| RF 18 | O sistema deve permitir qualquer piloto inscrito virar um piloto ADM | alta | nenhuma |

FONTE: OS AUTORES (2020)

* 1. REQUISITOS NÃO FUNCIONAIS

Os requisitos não funcionais representam características de qualidade que o sistema deve ter e que não estão relacionadas com suas funcionalidades.

As colunas possuem os seguintes significados para a TABELA 2

1. RNF – Identificação do Requisito, composto pela primeira letra da característica de qualidade mais numeração sequencial.
2. Descrição – descrição do requisito.

TABELA 02: REQUISITOS NÃO FUNCIONAIS

|  |  |
| --- | --- |
| RNF | Descrição |
| RNF 1 | O sistema deve ser feito em LOO java. |
| RNF 2 | O sistema deve usar o Postgre como banco de dados. |
| RNF 3 | IDE usadas: DBeaver e netbeans. |
| RNF 4 | O sistema deve ter uma interface gráfica usando swing. |
| RNF 5 | O tempo de resposta do software deve ser de um segundo para menos. |
| RNF 6 | O sistema é multiplataforma. |

FONTE: OS AUTORES (2020)

* 1. REGRAS DE NEGÓCIO

As regras de negócio são restrições/premissas necessárias para o negócio

“acontecer”.

As colunas possuem os seguintes significados para a TABELA 3:

1. RN – Identificação do Requisito, composto pela primeira letra da característica de qualidade mais numeração sequencial.

2. Descrição – descrição do requisito.

TABELA 03: REGRA DE NEGÓCIO

|  |  |
| --- | --- |
| RN | Descrição |
| RN 1 | Qualquer campeonato criado ao qual não tenha ao menos 8 pilotos não pode ocorre. |
| RN 2 | Nenhum piloto pode participar/criar de um campeonato caso o tal possua menos que 16 anos. |
| RN 3 | Deve possuir dois tipos de corridas: Campeonato e corridas normais. |
| RN 4 | Quando um campeonato ocorre, o piloto em 1° lugar aumenta sua Vitórias de competições. |
| RN 5 | Corridas só podem ser criadas em dias liberados pelo kartódromo. |
| RN 6 | A classificação dos pilotos deve ser baseada no tempo para completar as voltas. |
| RN 7 | Qualquer piloto que por algum motivo estiver sendo altamente negativo em suas ações é desclassificado e banido. |
| RN 8 | É proibido inscrição de cadastro de pilotos menores que 12 anos. |
| RN 9 | Permissão de apenas um campeonato por dia. |
| RN 10 | Permissão para desenvolvedores modificarem qualquer informação referente a pilotos e kartódromo. |
| RN 11 | O sistema deve ter um meio de punição via Strikes para pilotos que desrespeitarem/ofenderem/faltarem durante as corridas. |
| RN 12 | Aqueles que podem convidar pilotos para corridas são ADM do campeonato. |
| RN 13 | Os campeonatos devem classificar os competidores de acordo com suas pontuações. |
| RN 14 | Todo campeonato possui 4 voltas. |
| RN 15 | O ganhador de campeonato é aquele tiver a maior pontuação entre os competidores do campeonato. |

FONTE: OS AUTORES (2020)

* 1. TECNOLOGIAS UTILIZADAS

Para a criação deste projeto foi necessário algumas tecnologias para seu desenvolvimento, que vale citar cada uma delas abaixo na figura 4:

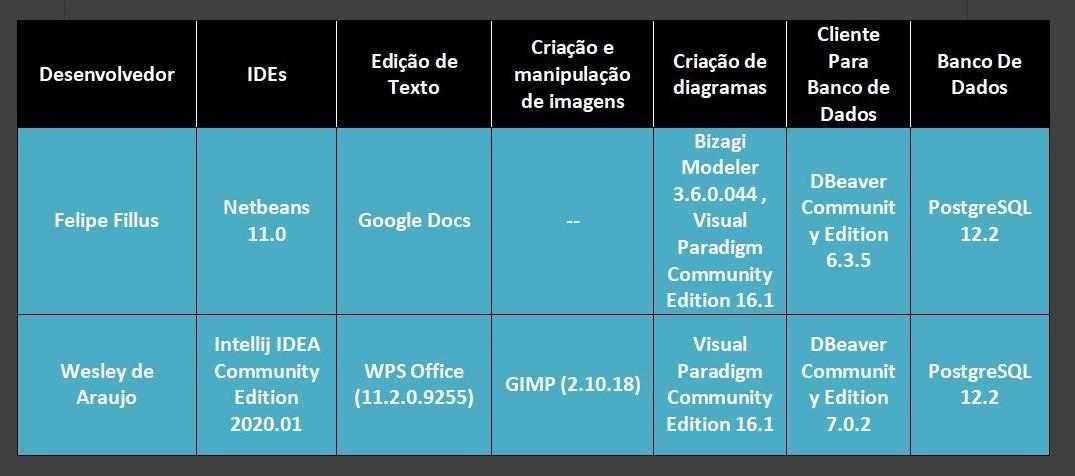


FIGURA 04: TECNOLOGIAS UTILIZADAS

FONTE: OS AUTORES (2020)

* 1. DIAGRAMA DE CASO DE USO

Um diagrama de caso de uso é uma visão gráfica de alguns ou todos os atores, casos de usos e seus relacionamentos identificados para um sistema. O diagrama descreve qual é a representação da fronteira do sistema (atores) e a função do sistema (casos de uso) (PIMENTEL, 2015).

No diagrama há também interações que são chamadas de relacionamento de associação, são representadas por linhas e servem para ligar os atores e os casos de uso, existem apenas 3 desses tipos de relações: <<include>>, <<extend>> e a generalização. Relacionamentos de <<include>> são criados entre um novo caso de uso e qualquer outro caso de uso que utilize esta funcionalidade, já um relacionamento de "extend" é usado para mostrar: comportamento opcional, executado sobre determinadas condições. Já a generalização ocorre quando há a relação entre um caso de uso C e um caso de uso D indicando que C é uma especialização de D. (PIMENTEL, 2015).

A figura 5 é referente ao diagrama de casos de uso do projeto:

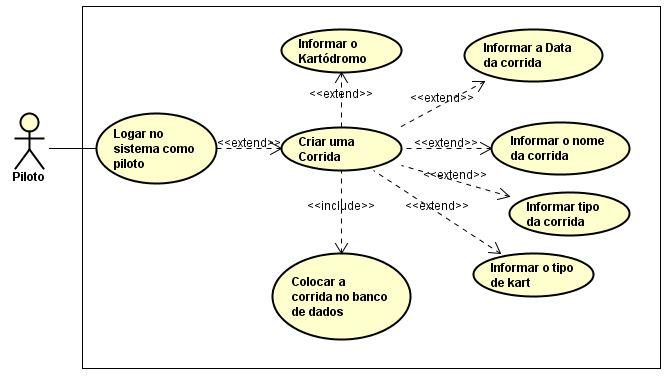


FIGURA 05: DIAGRAMA DE CASOS DE USO

FONTE: OS AUTORES (2020)

* 1. DESCRIÇÃO DE CASO DE USO

Um caso de uso deve descrever *o* queum sistema faz. Assim, é especificado o comportamento de um caso de uso pelo texto de seu fluxo de eventos, de modo que alguém de fora possa ler e compreender o que está especificando.

Quando se escreve o fluxo de eventos, deve-se incluir como e quando o caso de uso inicia e termina, quando o caso de uso interage com os atores (quem interage com o sistema) e outros casos de uso e quais são as informações transferidas e o fluxo básico e os fluxos alternativos do comportamento do sistema (PIMENTEL, 2015).

As colunas possuem os seguintes significados para a tabela:

1. Nome – Identificação do caso de uso.
2. Breve Descrição – descrição do caso de uso.
3. Fluxo de eventos – descrição dos eventos necessários para atingir o comportamento do caso de uso especificado.
4. Fluxos Alternativos – descrição mais detalhada dos eventos necessários do caso de uso.
5. Requisitos Especiais – condição especial para ocorrer o caso de uso.
6. Pré-condições – evento inicial para ocorrer o caso de uso.
7. Pós-condições – evento final para encerrar o caso de uso.
8. Pontos de extensão – evento necessário para ocorrer uma extensão do caso de uso.

TABELA 4 - DESCRIÇÃO DE CASO DE USO

|  |  |
| --- | --- |
| **Nome** | Criando uma corrida |
| **Breve**  **Descrição** | O piloto cria uma corrida em um kartódromo de sua escolha. |
| **Fluxo de eventos** | |
| O caso de uso começa com o piloto escolhendo criar uma corrida em seu perfil, ao qual informa o kartódromo onde a corrida será realizada,o tipo da corrida de seu interesse, a data que ocorrerá, o nome da corrida para os demais pilotos saberem do que se trata e o tipo de kart que vai ser usado. Após o piloto colocar as informações necessárias o piloto clica no botão de criar corrida para criar a corrida. | |
| **Fluxos alternativos** | |
| Caso o dia escolhido pelo piloto não for um dia da semana aberto pelo o kartódromo ou nesse dia já possuir uma corrida, a corrida não pode ser criada.  Caso o piloto não informar um dos campos para criar a corrida uma janela de mensagem aparece informando de falta de informações. | |
| **Requisitos Especiais** | Não possui. |
| **Pré-condições** | O piloto deve estar logado e a data escolhida deve estar aberta pelo kartódromo. |
| **Pós-condições** | A Corrida é inserida no banco de dados e o piloto que criou a corrida vira o administrador da corrida. |
| **Pontos de extensão** | Não se aplica. |

FONTE: OS AUTORES (2020)

* 1. DIAGRAMA DE ENTIDADE-RELACIONAMENTO (DER)

Um diagrama entidade relacionamento (ER) é um tipo de fluxograma que ilustra como “entidades”, p. ex., pessoas, objetos ou conceitos, se relacionam entre si dentro de um sistema. Este diagrama usa um conjunto definido de símbolos, tais como retângulos, diamantes, ovais e linhas de conexão para representar a conectividade entre as entidades, relacionamentos e seus atributos. Eles espelham estruturas gramaticais, onde entidades são substantivos e relacionamentos são verbos. (O QUE..., 2017).

A tabela 07 é referente ao DER do projeto:

TABELA 07: DIAGRAMA DE ENTIDADE-RELACIONAMENTO

img do DER(Bom, acho que essa vai demorar pra ficar na versao final já que pode mudar bastante ainda conforme nos fique desenvolvendo na codificação)

FONTE: OS AUTORES (2020)

* 1. DIAGRAMA DE CLASSES

Os diagramas de classes descrevem o que deve estar presente no sistema a ser modelado, são utilizados para representar os objetos que o sistema irá manipular, suas operações ou métodos. Os diversos componentes deste diagrama podem representar as classes que serão realmente programadas, os principais objetos ou as interações entre classes e objetos.

Resumidamente uma classe representa um objeto ou um conjunto de objetos que compartilham uma estrutura e comportamento comum. A forma de classe em si consiste em um retângulo com três linhas. A linha superior contém o nome da classe, a linha do meio, os atributos da classe e a linha inferior expressa os métodos ou operações que a classe pode utilizar. Um diagrama pode conter também o que é chamado de interação, que se refere às diversas ligações e relações que podem existir no diagrama. (O QUE..., 2018).

A tabela 08 é referente ao diagrama de classes do projeto:

TABELA 08: DIAGRAMA DE CLASSES

IMG DIAGRAMA(esse deixa pra colocar só mais pra frente quando estivermos fazendo a codificação final)

FONTE: OS AUTORES (2020)

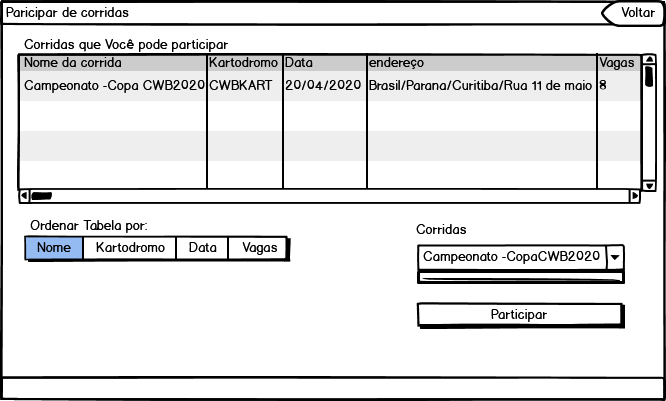
* 1. PROTÓTIPO

A prototipagem é basicamente uma versões incompletas do programa que está sendo desenvolvido, simulando alguns aspectos do programa, mas que no final pode ser completamente diferente do produto final uma vez que mudanças podem ocorrer.

A prototipagem de um programa tem o potencial de visualizar como o designer vai ficar e os criadores do software podem demonstrar aos seus usuários de como está ficando o programa já no início do projeto. Assim o cliente pode ver se projeto está correspondendo as especificações que deseja no software.

As tabelas 09 é referente a tela de inscrição do piloto em um corrida existente do projeto.

TABELA 09: PROTOTIPAGEM DA TELA DE INSCRIÇÃO DO PILOTO EM UMA CORRIDA



FONTE: OS AUTORES (2020)

**3. CONCLUSÃO**

O artigo teve a função de informar as funcionalidades e os objetivos deste projeto do sistema de gestão de kartódromo. Neste projeto, sistema feito poderá ser utilizado como uma opção para os clientes do kartódromo de criar suas corridas e convidar os demais pilotos para suas corridas.

Os objetivos propostos para a criação do sistema de gestão foram realizados com sucesso fazendo uso de métodos utilizados na programação orientada a objeto da linguagem Java, conseguindo sucesso na criação das telas, no desenvolvimento e na funcionalidade do sistema com o banco de dados.

**4. REFERÊNCIAS**

**ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. NBR 10520 – Informação e documentação**. Citação em documentos. Apresentação. Rio de

Janeiro, ago 2002a.

BUILDER, Project. **Tudo o que você precisa saber sobre o PM Canvas**. [*S. l.*], 03/08/2017. Disponível em: <https://www.projectbuilder.com.br/blog/tudo-o-que-voce-precisa-saber-sobre-o-pm-canvas/>. Acesso em: 19/04/2019.

BERTRAND, J. W. M.; FRANSOO, J. C. ***Modelling and simulation****: operations management research methodologies using quantitative modeling*. International Journal of Operations & Production Management, v. 22, n. 2, p. 241-264, 2002.

MAXIMIANO, A. C. A. **Teoria Geral da Administração**. 4. ed. São Paulo: Atlas, 2004.

**O QUE é um diagrama de classe UML?.** [*S. l.*: *s. n.*], 21/12/2018. Disponível em: https://www.lucidchart.com/pages/pt/o-que-e-diagrama-de-classe-uml. Acesso em: 29/08/2019.

**O QUE é um diagrama de entidade relacionamento?**. [S. l.: s. n.], 06/03/2017.

Disponível em: https://www.lucidchart.com/pages/pt/o-que-e-diagrama-entidade-

relacionamento. Acesso em: 17/09/2019.

OLIVEIRA NETTO, ALVIM ANTÔNIO DE; TAVARES, Wolmer Ricardo. **Introdução à engenharia de produção**. Florianópolis : Visual Books, 2006.

PIMENTEL, Andrey Ricardo. **Requisitos, Modelagem e UML**. [*S. l.*], 2015. Disponível em: <http://www.inf.ufpr.br/andrey/ci167/apostilaUml.pdf>. Acesso em: 11/08/2019.

VIEGAS, Gustavo. **Introdução a UML**. [*S. l.*], 2009. Disponível em: <https://www.devmedia.com.br/introducao-a-uml/6928>. Acesso em: 27/04/2019.