



Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Projeto Integrador II

Perfilômetro de Pista

Autor: Adolfo Serique, Antonino Martins, Geovanni de Jesus,
Guilherme Baldissera, José Rômulo Cordeiro, Karoline de
Oliveira, Lucas Barbosa, Miguel Pimentel, Pedro Calile, Pedro
Vinícius Moura, Vitor Umpierre

Orientador: Alex Reis, Guillermo Bestard, Rhander Viana,
Ricardo Chaim, Sébastien Rondineau

Brasília, DF

2019



Adolfo Serique, Antonino Martins, Geovanni de Jesus, Guilherme Baldissera,
José Rômulo Cordeiro, Karoline de Oliveira, Lucas Barbosa, Miguel Pimentel,
Pedro Calile, Pedro Vinícius Moura, Vitor Umpierre

Perfilômetro de Pista

(Relatório do Ponto de Controle 3 da disciplina de Projeto Integrador II)

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Alex Reis, Guillermo Bestard, Rhander Viana, Ricardo
Chaim, Sébastien Rondineau

Brasília, DF

2019

Listas de ilustrações

Figura 1 – Estrutura Analítica do Projeto	13
Figura 2 – Esquema de emissão e coleta de dados do sensor	19
Figura 3 – Parâmetros para o cálculo	20
Figura 4 – Distância mínima entre dois sensores	20
Figura 5 – Distância mínima entre dois sensores com margem de segurança	21
Figura 6 – MPU paralelo ao solo e rotacionado de 90 graus.	22
Figura 7 – Diagrama esquemático das conexões do MPU6050 com o ESP-12E e o servo motor.	23
Figura 8 – Diagrama esquemático da conexão do <i>VL53L0X</i> com o <i>Attiny85</i>	24
Figura 9 – Diagrama esquemático da conexão do <i>GY-NEO6MV2</i> com a <i>Raspberry PI</i>	26
Figura 10 – Diagrama esquemático do teste da memória <i>flash</i> do <i>ESP-12E</i>	27
Figura 11 – Barramento <i>I²C</i>	28
Figura 12 – Sistema de transmissão UART.	30
Figura 13 – Sistema de transmissão SPI.	31
Figura 14 – <i>Layout</i> recomendado do GND	35
Figura 15 – Gerador de energia	37
Figura 16 – Relação a polia e o pneu	39
Figura 17 – Circuito do carregador de bateria	40
Figura 18 – Reboque utilizado	43
Figura 19 – Vista Isométrica	44
Figura 20 – Amortecedor	45
Figura 21 – Cantoneira longitudinal	46
Figura 22 – Cantoneira transversal	47
Figura 23 – Coluna frontal	48
Figura 24 – Coluna frontal	49
Figura 25 – Chapa dos sensores	50
Figura 26 – Chapa dos sensores	51
Figura 27 – Lança direita	52
Figura 28 – Lança esquerda	53
Figura 29 – Longarina direita	54
Figura 30 – Longarina esquerda	55
Figura 31 – Parachoque	56
Figura 32 – Paralama	57
Figura 33 – Perfil em Z	58
Figura 34 – Tubo circular no cambão	59
Figura 35 – Vigas	60

Figura 36 – Vigas dos sensores	61
Figura 37 – Perfil U enrijecido	63
Figura 38 – Perfil circular	63
Figura 39 – Cortes realizados na chapa	64
Figura 40 – Malha estruturada	65
Figura 41 – Força distribuída aplicada	65
Figura 42 – Carregamento distribuída aplicado	66
Figura 43 – Concentrador de tensão do cambão e o quadro	66
Figura 44 – Tensões equivalentes utilizando o critério de Von Mises	67
Figura 45 – Diagrama de Classe - Versão 0.1	69
Figura 46 – Estrutura de Cena - VIP	70
Figura 47 – Diagrama de Classe - Versão 1.0	70
Figura 48 – Micro Serviços - Versão 1.0	71
Figura 49 – Micro Serviços - Versão 2.0	72
Figura 50 – Diagrama de caso de uso	73
Figura 51 – Diagrama de componentes	73
Figura 52 – Tela inicial do aplicativo do perfilômetro	77
Figura 53 – Tela inicial de traçar novo perfil	78
Figura 54 – Feedback traçando perfil	79
Figura 55 – Lista de perfis	80
Figura 56 – Ver perfil	81
Figura 57 – Tela de comparação de perfis	82
Figura 58 – Tela listando sensores	83
Figura 59 – Feedback de calibração dos sensores	84
Figura 60 – Avisos ao usuário	85
Figura 61 – Gráfico de riscos	89

Listas de tabelas

Tabela 1 – Tabela de integrantes	14
Tabela 2 – Parâmetros retirados das especificações I^2C	29
Tabela 3 – Teste de Velocidade x Potência	38
Tabela 4 – Backlog atual das atividades	86
Tabela 5 – Milestones	87
Tabela 6 – Custos do projeto	88

Sumário

1	OBJETIVOS	11
1.1	Objetivos Gerais	11
1.2	Objetivos Específicos	11
2	ESPECIFICAÇÕES	13
2.1	Estrutura Analítica do Projeto (EAP)	13
2.2	Equipe	14
2.3	Objetivos específicos	14
2.3.1	Eletrônica	14
2.3.2	Energia	14
2.3.3	Estrutura	15
2.3.4	Software	15
2.4	Requisitos para o projeto	16
2.4.1	Eletrônica	16
2.4.2	Energia	16
2.4.3	Estrutura	17
2.4.4	Software	17
3	SUBSISTEMAS	19
3.1	Eletrônica	19
3.1.1	Cálculo da distância mínima entre os sensores <i>laser</i>	19
3.1.2	Cálculo da velocidade máxima do reboque	21
3.1.3	Testes	21
3.1.3.0.1	MPU6050	21
3.1.3.0.2	VL53L0X	24
3.1.3.0.3	GY-NEO6MV2	24
3.1.3.0.4	Sistema de arquivo do <i>ESP-12E</i>	26
3.1.3.1	Implementação eletrônica	27
3.1.3.1.1	Barramento <i>I²C</i>	28
3.1.3.1.2	Barramento <i>UART</i>	29
3.1.3.1.3	Barramento <i>SPI</i>	30
3.1.3.1.4	Módulo <i>GPS</i>	31
3.1.3.1.5	Módulo sensor <i>laser</i>	32
3.1.3.1.6	Módulo Acelerômetro/Giroscópio	34
3.1.3.2	Arquitetura da Placa de Circuito Impresso	35
3.2	Energia	37

3.2.1	Gerador de energia	37
3.2.1.1	Dínamo	37
3.2.1.2	Polia	38
3.2.2	Recarga da bateria	39
3.2.2.1	Bateria 12V 7Ah	40
3.2.2.2	Regulador de tensão DC-DC	40
3.3	Estrutura	42
3.3.1	Estrutura do Reboque	42
3.3.1.1	Desenho Técnico	43
3.3.1.2	Material Utilizado	61
3.3.1.3	Modelo Analítico	62
3.3.1.4	Modelo de Elementos Finitos	62
3.3.1.5	Análise estrutural - Modelo de Análise	63
3.3.1.6	Simulação	64
3.4	Software	68
3.4.1	Tecnologias Utilizadas	68
3.4.2	Diagramas	69
3.4.2.1	Diagrama de Classes	69
3.4.2.1.1	Diagrama de Classe - MVP - Model View Presenter (0.1)	69
3.4.2.1.2	Diagrama de Classe - VIP - View Interactor Presenter (1.0)	70
3.4.2.2	Microserviços e Frameworks	71
3.4.2.2.1	Versão 1.0	71
3.4.2.2.2	Versão 2.0	72
3.4.2.3	Diagrama de Caso de uso	73
3.4.2.4	Diagrama de componentes	73
3.4.3	Arquitetura Backend	74
3.4.4	Arquitetura Framework Gráfico em três Dimensões	74
3.4.5	Política de Branches - GitFlow	74
3.4.6	Papéis de Atuação	75
3.4.6.1	Scrum Master	75
3.4.6.2	DevOps	75
3.4.6.3	Arquiteto	76
3.4.7	Protótipo de Papel	76
3.4.8	Atividades Realizadas	85
4	GERENCIAMENTO DO PROJETO	87
4.1	Cronograma e Sequenciamento de Atividades	87
4.2	Milestones Identificados	87
4.3	Custos atuais do projeto	87
4.4	Riscos	88

REFERÊNCIAS	91
ANEXO A – EAP DETALHADA	93
ANEXO B – ESQUEMÁTICO ELÉTRICO GERAL	95
ANEXO C – CÓDIGO DA ESP-12E PARA TESTE DO MPU	97
ANEXO D – CÓDIGO DA ESP-12E PARA TESTE DE MEMÓRIA	101
ANEXO E – CÓDIGO DO ATTINY85 PARA TESTE DO LASER	105

Contextualização

Ao longo dos últimos anos, foi possível notar a importância da medida da irregularidade longitudinal dos pavimentos no setor rodoviário, obtendo assim uma avaliação das condições funcionais de uma via.

A irregularidade longitudinal, medida de acordo com os dispositivos de tipo resposta, é o somatório dos desvios da superfície de um pavimento em relação a um plano de referência ideal de projeto geométrico que afeta a dinâmica do veículo, o efeito dinâmico das cargas, a qualidade ao rolamento e a drenagem superficial da via. O dispositivo tem grande utilização em veículos nacionais em função do custo e facilidade no processo de obtenção de dados. No entanto, não é o mais utilizado, pois não oferece grande precisão das irregularidades de pavimento.

Medições dinâmicas do perfil da estrada são geralmente feitas com instrumentos montados em veículos como os perfilômetros a laser. A abordagem consiste em um sensor que mede a altura do veículo em relação à estrada. Um acelerômetro é duplamente integrado para fornecer a altura do sensor em relação ao ponto de partida. A diferença entre os dois é o perfil de elevação da estrada. Este perfil de elevação é então processado para obter a irregularidade da pista.

Aspectos importantes na análise do dispositivo do tipo medidor de perfil (perfilômetro) são levados em consideração como a sensibilidade do equipamento em função ao nível de irregularidade do pavimento de acordo com o valor do QI (fator de irregularidade longitudinal da pista) o valor deste fator medido em milímetros tem como referência a sensibilidade em que o equipamento está calibrado para que os dados possam ser estimados com uma precisão maior.

Portanto, em relação aos argumentos explicitados acima, este trabalho tem como finalidade principal a análise das informações obtidas e armazenadas nos bancos de dados dos perfis de irregularidade de pistas com a utilização de um perfilômetro a laser.

1 Objetivos

1.1 Objetivos Gerais

Desenvolver um equipamento que trace o perfil de uma pista, por meio de sensores eletrônicos a laser que faça uma análise de maneira que o usuário do equipamento possa interpretar os dados colhidos e que qualquer veículo que possua uma engate para reboque seja capaz de utiliza-lo.

1.2 Objetivos Específicos

- Elaboração do projeto estrutural;
- Elaboração do sistema autônomo de alimentação;
- Elaboração do sistema de aquisição de dados;
- Elaboração do sistema eletrônico;
- Desenvolver aplicativo que interprete os dados adquiridos e os mostre para o usuário;
- Integração dos sistemas.

2 Especificações

O projeto em questão tem como princípio fundamental a construção de um perfilômetro de pista auto sustentável que utiliza sensoriamento a laser e busca traçar e especificar as irregularidades dos pavimento presentes em vias.

O equipamento deve ser conectado a um veículo com engate pra reboque, pois o equipamento será uma carretinha. Para o seu funcionamento de traçar o perfil da pista, ele não deve ser utilizado com tempo chuvoso ou com pista molhada. Os equipamentos devem ser alimentados energeticamente sem a necessidade de fontes externas de energia. Para o dispositivo ser acionado deve haver um aplicativo responsável por iniciar o processo de gerar o perfil, esse aplicativo será responsável por dar opções ao usuário do que é possível fazer com o dispositivo.

2.1 Estrutura Analítica do Projeto (EAP)

A EAP do projeto mateve-se a mesma desde a criação do projeto. As atividades chaves nos pontos de controle andaram conforme o processo e sempre mantendo a parte de gerência em alta. Ela também está no Anexo A.

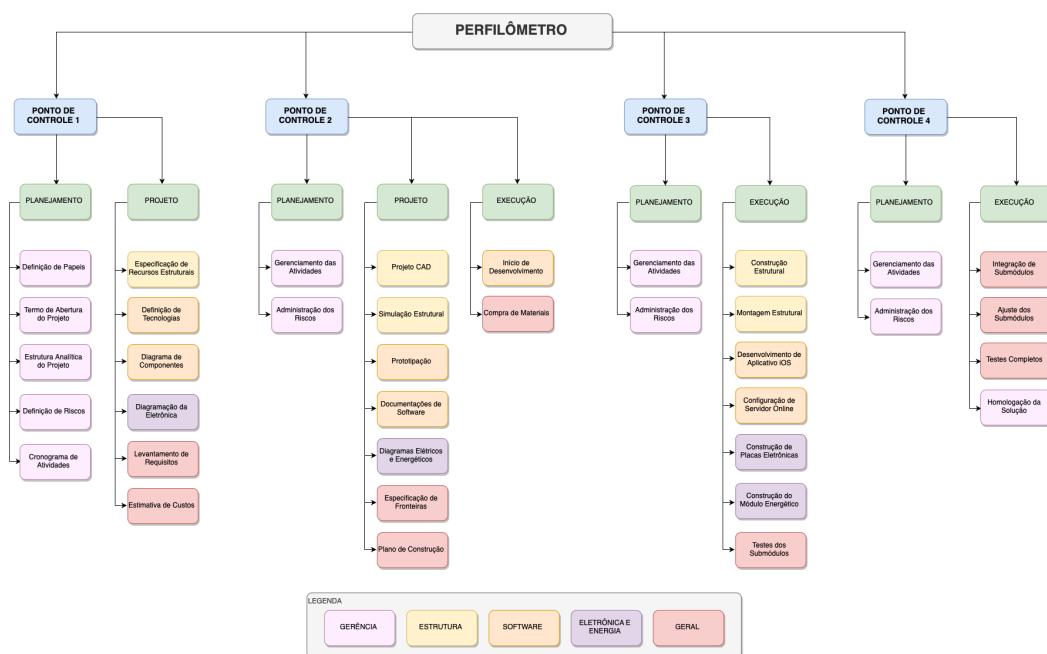


Figura 1 – Estrutura Analítica do Projeto

2.2 Equipe

A equipe se manteve com os mesmos papéis e cargos até o momento. Segue mais detalhes:

Integrante	Engenharia	Cargos de Atuação
Adolfo Serique	Eletrônica	Diretor de Qualidade, Diretor Eletrônico
Antonino Martins	Aeroespacial	Coordenador Geral, Diretor Aeroespacial
Geovanni de Jesus	Software	Desenvolvedor
Guilherme Baldissera	Software	Diretor de Software
José Rômulo Cordeiro	Aeroespacial	Desenvolvedor
Karoline de Oliveira	Energia	Desenvolvedor
Lucas Barbosa	Eletrônica	Desenvolvedor
Miguel Pimentel	Software	Desenvolvedor
Pedro Vinícius Moura	Aeroespacial	Desenvolvedor
Pedro Calile	Eletrônica	Desenvolvedor
Vitor Umpierre	Energia	Desenvolvedor

Tabela 1 – Tabela de integrantes

2.3 Objetivos específicos

2.3.1 Eletrônica

- O perfilômetro será composto por sensores de distância *laser* que serão capazes de geral o perfil da pista pelas distâncias medidas pelos sensores;
- Os sensores serão ajustados por meio de quatro servomotores, de acordo com o ângulo medido pelo giroscópio, com o objetivo de ficarem paralelos a pista;
- O acelerômetro será responsável pelo referencial inercial, de tal modo a medir oscilações geradas pelo reboque, e desconta-las na medida dos sensores *laser*, para assim, o perfil medido ser o mais preciso possível;
- O módulo *GPS* será responsável por traçar a rota realizada pelo reboque;
- O microcontrolador conterá um módulo *Bluetooth* para a comunicação com o aplicativo.

2.3.2 Energia

- O protótipo deverá ser autônomo, sendo assim, não dependerá de nenhum fornecimento de energia advindo do veículo ao qual está acoplado;

- A fonte de alimentação utilizada deverá ser suficiente para manter o sistema em pleno funcionamento durante todo o percurso;
- A geração de energia será feita por meio de energia mecânia, advinda da rotação das rodas do protótipo e assim será fornecida carga para uma bateria automotiva;
- Todo o sistema deverá ser protegido de danos que podem ser causados pelo meio em que o reboque esta sendo utilizado ou por efeitos vibratórios, dentre outros.

2.3.3 Estrutura

- O perfilômetro deverá estar acoplado em um reboque que poderá usado em qualquer carro. O reboque deve ser projetado para que suporte todas os esforços requeridos pela estrutura e carga útil.
- Deve-se procurar uma altura mínima para que os sensores atuem de maneira eficaz.
- O quadro de suspensão deverá ser baseada no que é utilizado no antigo modelo Fusca. Este sistema possibilita a regulagem de altura e nivelamento independente para os sensores.
- A estrutura deve ser feita de maneira que reduza os efeitos vibratórios.
- A estrutura deverá oferecer suporte para os sistemas de alimentação de energia e para os materiais eletrônicos.
- Escolha do material adequada para cada parte da estrutura com custo benefício acessível.

2.3.4 Software

No processo de concepção do projeto foram elaborados diversos requisitos que podem ser associados à área de software. Destes, muitos estão relacionados a forma de obter, processar e disponibilizar os dados provindos do Perfilômetro. Desta forma, a seguir estão descritos mais detalhadamente estes requisitos.

- O usuário deverá ser capaz de analisar os dados obtidos através de um aplicativo;
- Os dados deverão ser obtidos através dos sensores;
- Os dados deverão ser salvos em um banco de dados remoto;
- O percurso feito pelo carrinho deverá ser mostrado para o usuário;

- Os dados deverão ser mostrados para o usuário em um gráfico que mostre os níveis de ondulação na pista e seja interativo, ou seja para cada ponto obtido, o seu local no mapa deverá ser indicado para o usuário;
- O aplicativo deverá ser capaz de gerenciar o perfilômetro, como ajustar a calibragem dos sensores, iniciar o funcionamento dos sensores para traçar o perfil da pista e gravar o trajeto;
- O aplicativo deverá estar disponível na plataforma iOS;
- O celular deverá ter acesso à internet para mandar os dados obtidos para o banco de dados remoto;
- O aplicativo deverá ter um banco de dados local para armazenar os dados caso o celular não tenha internet para mandar os dados obtidos para o banco remoto, e mandá-los assim que obtiver acesso à uma rede de internet.

2.4 Requisitos para o projeto

2.4.1 Eletrônica

- Ser composto por sensores de distância laser;
- Os sensores deverão ser ajustados por dois servomotores;
- O referencial inicial deverá ser provido por um acelerômetro;
- A rota realizada pelo reboque deve ser traçada por um módulo GPS;
- O microcontrolador deve conter um módulo Bluetooth para comunicação com o aplicativo.

2.4.2 Energia

- O sistema de alimentação deve sustentar o sistema por meios próprios;
- A fonte de alimentação deve ser capaz de manter o sistema durante o uso do equipamento;
- A alimentação deve ser feita por energia mecânica;
- O sistema de alimentação deve estar protegido contra danos e vibrações do reboque.

2.4.3 Estrutura

- O perfilômetro deve estar acoplado em um reboque;
- O perfilômetro deve ter uma altura de aproximadamente 15 cm;
- A suspensão deve poder ter sua altura regulada;(suspensão semelhante à do fusca)
- A estrutura deve reduzir os efeitos vibratórios;
- A estrutura deve oferecer suporte para o sistema de alimentação de energia para os dispositivos eletrônicos.

2.4.4 Software

- Um relatório do percurso feito pelo equipamento deve ser gerado para ser analisado;
- Os dados para gerar o relatório devem ser obtidos através dos sensores;
- Os dados deverão ser sincronizados com um banco remoto;
- O percurso feito pelo engate deve ser mostrado para o usuário;
- Os dados deverão ser mostrado com um grafico mapeando os níveis de ondulação da pista;
- O aplicativo deverá ser capaz de calibrar a altura dos sensores;
- O aplicativo deve estar disponível para plataforma iOS;
- O celular deve estar conectado à internet para o uso do Banco de dados Remoto.

3 Subsistemas

3.1 Eletrônica

O objetivo consiste na análise do comportamento dos sensores em relação a estrutura e a velocidade do perfilômetro. Deve-se calcular a distância mínima que um sensor pode ficar de outro e a velocidade máxima do reboque, utilizando os dados obtidos nos requisitos do projeto. Além disto, foram realizados testes nos sensores, para se estudar melhor o seu comportamento no sistema como um todo.

3.1.1 Cálculo da distância mínima entre os sensores *laser*

Para o cálculo dos parâmetros utilizados no projeto, utilizou-se o *datasheet* do *VL53L0X* como referência, tal como mostra a Fig. 2 a seguir.

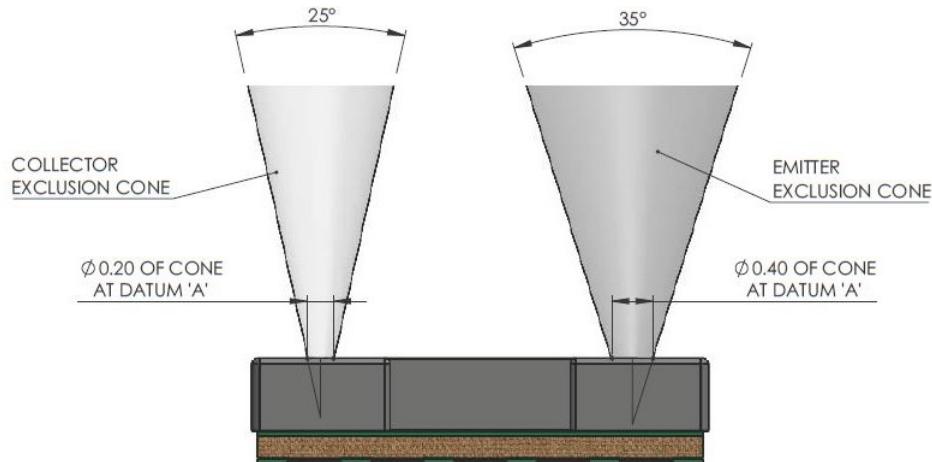


Figura 2 – Esquema de emissão e coleta de dados do sensor

Para calcular a distância mínima entre dois sensores, considerando uma altura inicial h , os ângulos e as larguras L_c (diâmetro do coleto) e L_e (diâmetro do emissor), todos os dados em relação a altura h , como mostra a Fig. 3.

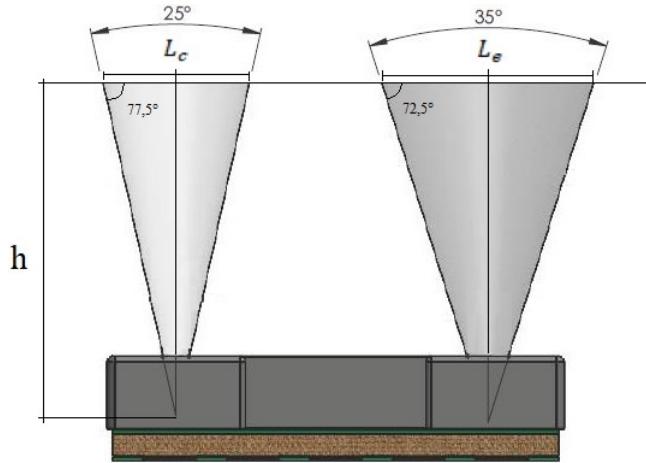


Figura 3 – Parâmetros para o cálculo

Assim, os diâmetros do emissor e do coletor podem ser calculados da seguinte forma:

$$\tan(77.5^\circ) = \frac{h}{\frac{L_c}{2}} \Rightarrow L_c = \frac{2 \times h}{\tan(77.5^\circ)} \quad (3.1)$$

$$\tan(72.5^\circ) = \frac{h}{\frac{L_e}{2}} \Rightarrow L_e = \frac{2 \times h}{\tan(72.5^\circ)} \quad (3.2)$$

Com estes dados, a distância mínima entre os sensores pode ser calculada somando os raios do emissor e do coletor, como está representado na Fig. 60 pela variável r .

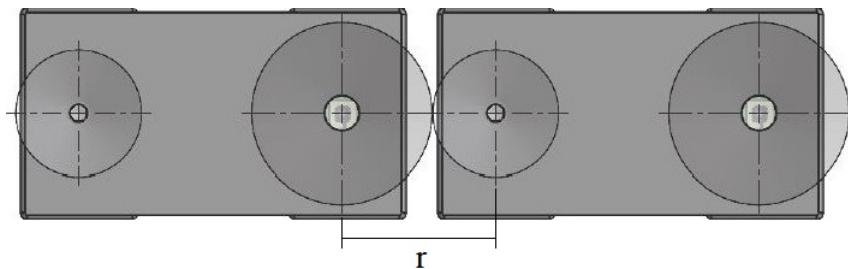


Figura 4 – Distância mínima entre dois sensores

O cálculo de r é realizado da seguinte forma:

$$r = \frac{\frac{2 \times h}{\tan(72.5^\circ)} + \frac{2 \times h}{\tan(77.5^\circ)}}{2} = h \times \left(\frac{\tan(77.5^\circ) + \tan(72.5^\circ)}{\tan(77.5^\circ) \times \tan(72.5^\circ)} \right) \Rightarrow r \cong h \times 0.537 \quad (3.3)$$

Para se evitar um erro de medida adiciona-se uma variável de segurança m entre os sensores, assim as medidas irão possuir maior confiabilidade. A Fig. 5 ilustra a distância mínima entre dois sensores de forma segura.

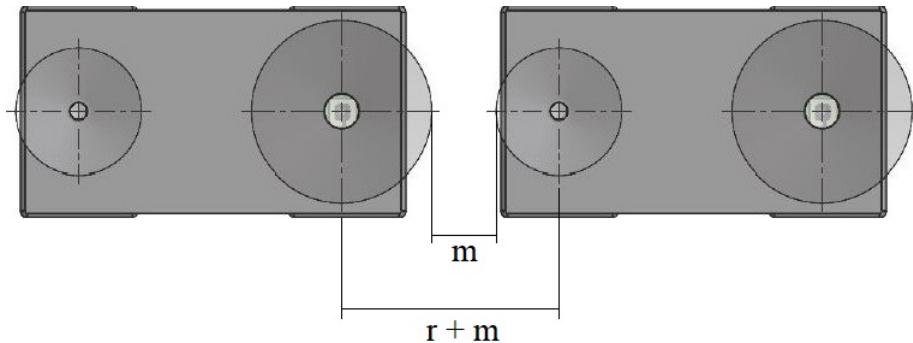


Figura 5 – Distância mínima entre dois sensores com margem de segurança

3.1.2 Cálculo da velocidade máxima do reboque

Como o objetivo de diminuir o tempo entre duas medidas($20\ ms$) do sensor *laser*, fileiras de sensores foram implementadas. Para cada fileira, o tempo de $20\ ms$ é reduzido pela metade. Para realizar o cálculo da velocidade do reboque(v), tomou-se as seguintes variáveis encontradas do escopo do projeto: t (*timing budget* ou tempo entre duas medidas do sensor), n (número de fileiras) e w (distância entre duas medidas). As variáveis se relacionam da seguinte forma:

$$v = \frac{2^{(n-1)} \times w}{t} \quad (3.4)$$

Onde t é dado em milissegundos (ms), w é dado em milímetros(mm) e v é calculado em metros por segundo(m/s).

3.1.3 Testes

Com o objetivo de analisar cada subsistema separadamente foram realizados testes dos sensores especificados para entender seu comportamento e facilitar na fase de integração com as outras engenharias.

3.1.3.0.1 MPU6050

O MPU6050 é um dispositivo capaz de medir força e velocidade, ou seja, possui um acelerômetro e um giroscópio, assim sendo, o MPU não mede ângulos diretamente, para isso requer alguns cálculos.

O acelerômetro mede aceleração, e essa aceleração pode ser expressa em três eixos, os eixos X, Y e Z. Pelo fato do MPU detectar a aceleração da gravidade da terra, é possível graças a gravidade terrestre determinar o ângulo de inclinação em relação ao eixo X e Y. Supondo que o MPU esteja paralelo ao solo, como na Fig. 6, o eixo Z marcará 9.8 enquanto os demais eixos marcarão 0. Ao girar o MPU 90 graus, o eixo X é quem

marcará 9.8. Sabendo que a aceleração da gravidade é de $9.8m/s^2$ e tendo como base as considerações feitas acima, por trigonometria é possível calcular o ângulo de inclinação do MPU.

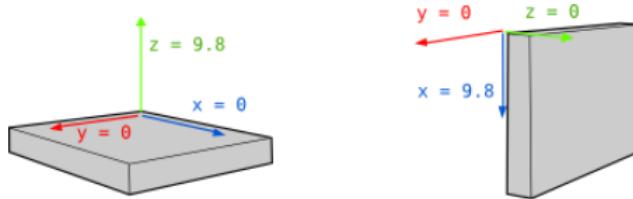


Figura 6 – MPU paralelo ao solo e rotacionado de 90 graus.

Uma forma para calcular os ângulos X e Y é por meio da tangente, como pode ser visto nas equações (3.5) e (3.6), respectivamente.

$$\text{angulo}Y = \arctan(x/\sqrt{y^2 + z^2}) \quad (3.5)$$

$$\text{angulo}X = \arctan(y/\sqrt{x^2 + z^2}) \quad (3.6)$$

O giroscópio mede a velocidade angular, ou seja, mede o número de graus girados em um segundo. Conhecendo o ângulo inicial do MPU, pode-se adicionar o valor que marca o giroscópio para conhecer o novo ângulo a cada momento. Assim sendo, ao considerarmos que o MPU comece a 0 graus, se o giroscópio fizer uma medição a cada intervalo de tempo e marcar um certo valor no eixo Y, o ângulo pode ser obtido por meio da seguinte equação:

$$\text{angulo}Y = \text{angulo}Y_{\text{anterior}} + \text{giroscopio}Y \cdot \Delta T. \quad (3.7)$$

Onde ΔT é o tempo que passa cada vez que esta fórmula é calculada, $\text{angulo}Y_{\text{anterior}}$ é o ângulo calculado na última vez que esta fórmula foi chamada e $\text{giroscopio}Y$ é a leitura atual do ângulo Y do giroscópio.

As leituras reais de qualquer sensor apresentam ruídos e erros, o que não foge ao caso para o *MPU6050*. Para tentar amenizar os erros e ruídos do sensor, aplicou-se um filtro complementar, que consiste basicamente em um filtro passa-alto para o giroscópio e um passa-baixo para o acelerômetro. A combinação desses dois filtros é que resulta no filtro complementar, que pode ser visto pela equação a seguir, que vale tanto para o ângulo X, quanto para o ângulo Y.

$$\text{angulo} = 0.98 \cdot (\text{angulo} + \text{angulo}Giroscopio \cdot \Delta T) + 0.02 \cdot \text{anguloAcelerometro} \quad (3.8)$$

Na prática, esse algoritmo foi implementado em uma ESP-12E obedecendo as seguintes conexões:

- MPU VCC --> ESP-12E 3v3
 - MPU GND --> ESP-12E GND
 - MPU SCL --> ESP-12E D2
 - MPU SDA --> ESP-12E D1
 - MPU AD0 --> ESP-12E GND

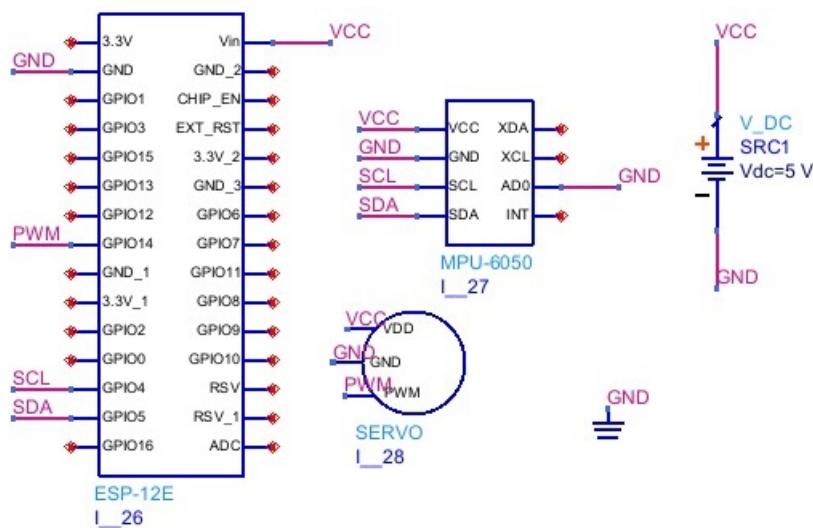


Figura 7 – Diagrama esquemático das conexões do MPU6050 com o ESP-12E e o servo motor.

Como os valores obtidos pelo sensor para cada eixo eram valores brutos, foi necessário olhar no *datasheet* do fabricante para determinar a sensibilidade tanto do acelerômetro quanto para o giroscópio, de forma que ao dividir os valores brutos pela sensibilidade, obtém-se os ângulos em graus, valores estes adequados para se tomar as decisões cabíveis na aplicação do projeto. O algoritmo utilizado para este teste encontra-se no Anexo C.

Os ângulos obtidos pelo filtro complementar serão usados para controle de um servo motor responsável por corrigir o posicionamento inicial do sensor a laser que será descrito posteriormente.

O servo motor utilizado foi um micro servo 9g SG90, cujas conexões com a *ESP-12E* podem ser vistas na Fig. 7 e o algoritmo usado também no Anexo C, obedecendo as seguintes conexões:

- SERVO VERMELHO ==> ESP Vin

- SERVO PRETO --> ESP GND
- SERVO AMARELO --> ESP D5

3.1.3.0.2 VL53L0X

O módulo *laser* *VL53L0X* foi ligado usando um *Attiny85*, assim conseguiu-se testar o seu funcionamento. Notou-se que o tempo entre duas medidas era de 33 ms , assim como se esperava ao analisar o *datasheet* do sensor. Com alguns testes no sensor, conseguiu-se diminuir o tempo para 20 ms (o mínimo suportado pelo sensor). A Fig. 8 mostra o esquema de ligação entre o sensor e o controlador, tal como está descrito a seguir (o algoritmo encontra-se no Anexo E):

- VL53L0X VCC --> ATTINY 5 V
- VL53L0X GND --> ATTINY GND
- VL53L0X SDA --> ATTINY SDA
- VL53L0X SCL --> ATTINY SCL

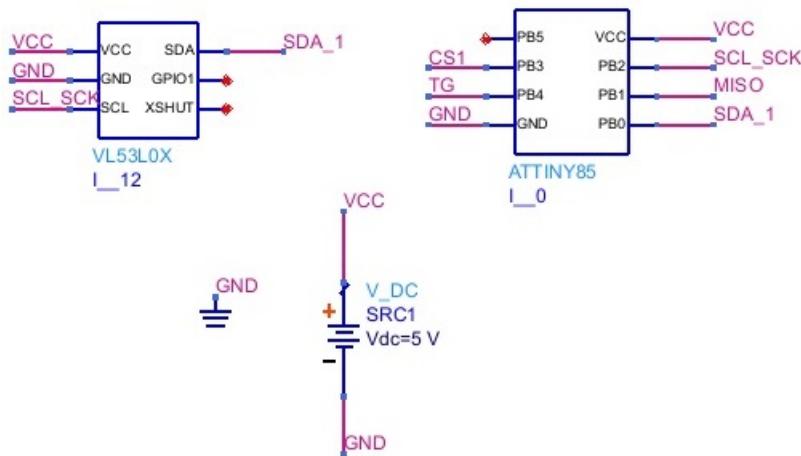


Figura 8 – Diagrama esquemático da conexão do *VL53L0X* com o *Attiny85*.

3.1.3.0.3 GY-NEO6MV2

O módulo de radionavegação por satélite *GPS NEO6MV2* necessita de apenas 4 conexões: *RX*, *TX*, *Vcc* e *GND*, o que torna-o bastante simples de incorporar com o uso do conector *UART* na *Raspberry PI*, utilizando diretamente o conversor de nível lógico bidirecional 3v3 capaz de fazer um *step-down* ou *step-up* entre as tensões de $3.3V$ e $5V$.

A *Raspberry PI* usa, como padrão, o *UART* como um console serial, por isso precisa-se desativar esta funcionalidade para que possa-se usar o *UART* para a nossa aplicação. Após abrir uma sessão de terminal na *Raspberry PI*, é necessário fazer o backup do arquivo *cmdline.txt* antes de editá-lo com o comando (*sudo cp /boot/cmdline.txt /boot/cmdline_backup.txt*). Então edita-se o arquivo removendo a interface serial com o comando (*sudo nano /boot/cmdline.txt*) e deleta-se usando o comando (*console = ttymAMA0, 115200*) e, por fim, salvando o arquivo com o comando (*ctrl X, Y*). Em seguida, digita-se no terminal o comando (*sudo nano/etc/inittab*), entrando *ttymAMA0* e pressionando o comando de escrita (*ctrl W*) e digitando *ttymAMA0* no campo de pesquisa. Quando encontra-se a linha, pressiona-se *home*, insere-se o símbolo *#* para comentar a linha e o comando (*ctrl X, Y*) para salvar. Por fim digite o comando (*sudo reboot*) para reiniciar a *Raspberry PI*.

A Fig. 9 mostra o esquema de ligação entre o sensor e a *Raspberry Pi*, tal como está descrito a seguir:

- GY-NEO6MV2 VCC --> Raspberry PI 5 V
- GY-NEO6MV2 GND --> Raspberry PI GND
- GY-NEO6MV2 TX --> Raspberry PI GPIO 14
- GY-NEO6MV2 RX --> Raspberry PI GPIO 15

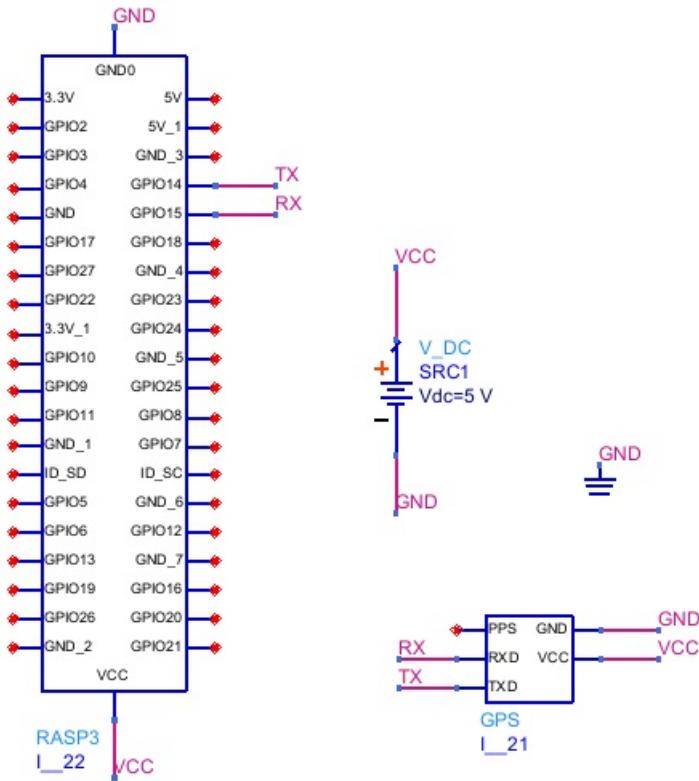


Figura 9 – Diagrama esquemático da conexão do *GY-NEO6MV2* com a *Raspberry PI*.

3.1.3.0.4 Sistema de arquivo do *ESP-12E*

Para fins deste projeto, é essencial o armazenamento dos dados coletados pelos sensores para que assim estes possam ser processados ao fim de cada procedimento. Uma forma de atender essa demanda é criar um sistema de *log* em arquivo utilizando o *SPIFFS* (*SPI Flash File System*).

O *SPIFFS* foi desenhado para sistemas com memória *SPI NOR flash*, em dispositivos embarcados. O sistema de arquivos tem como principal meta, usar o mínimo possível de memória *RAM*. Um fator limitante deste sistema é que ele não suporta pastas, porém o nome do arquivo pode conter “barra”, como por exemplo “/log/arquivo.txt” e subentende-se que o arquivo está dentro da pasta *log*, por convenção.

O tamanho do sistema de arquivos depende do tamanho do *chip flash*. A *ESP-12E* possui um chip flash com 4 *MB* de mória, enquanto que apenas 3 *MB* é reservado para o sistema de arquivos. Para que essa memória seja acessada, é necessário a inclusão no código fonte da biblioteca *FS.h*.

A avaliação da memória, consistiu em analisar quantos *bytes* por segundo pode-se gerar com os sensores. Como o sensor *laser* gera 3 *bytes* a cada 20 *ms*, um cálculo rápido foi realizado para avilar quanto tempo até a memória ficar cheia.

$$20 \text{ ms} = 50 \text{ medidas por segundo} \Rightarrow 50 \times 3 = 150 \text{ bytes/segundo} \quad (3.9)$$

Com uma regra de 3 simples, determina-se quantos segundos a *ESP-12E* consegue ficar ligada sem lotar a sua memória de 3 MB, onde x é o tempo a se determinar.

$$150 \times x = 3 \times 10^6 \Rightarrow x = 2 \times 10^4 \text{ segundos} \quad (3.10)$$

Por tanto, a memória é suficiente para a aplicação do projeto.

Para testar esse sistema, pretendia-se gravar dados permanentes na memória *flash* da *ESP-12E* e analisar o seu comportamento. Com isso, foi desenvolvido um sistema em que liga/desliga um *LED* e salva o seu estado na memória. O estado ficará salvo mesmo que a ESP seja desligada, de forma que ao ser ligada novamente o sistema busca na memória o estado do *LED* acionando-o de acordo com seu estado antes da placa ser desligada. As conexões realizadas para esse procedimento podem ser vistas na Fig. 10.

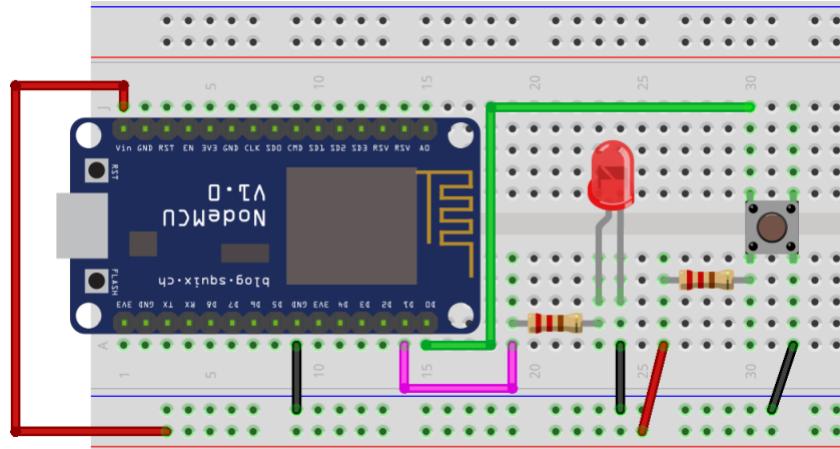


Figura 10 – Diagrama esquemático do teste da memória *flash* do *ESP-12E*.

O código foi implementado e verificado o funcionamento da memória *flash*, constatando então que a mesma pode ser usada para gravar os dados dos sensores. O código fonte do teste feito pode ser visto no Anexo D.

PARTE QUE VEIO DE OUTRA SEÇÃO DO PROJETO, REVISAR!

3.1.3.1 Implementação eletrônica

Na implementação do projeto serão utilizados dois protocolos de comunicação, *I²C* (*Inter Integrated Circuit*) e *UART* (*Universal Asynchronous Receiver Transmitter*). Para o protocolo *I²C* será utilizado um esquema de barramento que permite a conexão de uma grande quantidade de sensores em uma mesma rede, assim sendo, por meio deste

protocolo serão controlados os sensores a *laser* e acelerômetro/giroscópio. Já o módulo *GPS* (*Global Positioning System*), utiliza comunicação serial e apenas dois pinos (*RX* e *TX*), então o protocolo *UART* será adotado pela sua simplicidade e sistema *full-duplex*.

3.1.3.1.1 Barramento I^2C

I^2C é um protocolo de comunicação de barramento serial que utiliza apenas dois fios, inventada na déca de 90 pela *Philips* para conectar periféricos de baixa velocidade a placas-mãe, microcontroladores e equivalentes.

O barramento é composto por dois fios, *SDA* (*Serial Data*) e *SCL* (*Serial Clock*) e alimentação (*VDD*) típicamente de 3.3 V ou 5 V. Os fios de comunicação possuem *pull-ups*, como pode ser visto na figura 11.

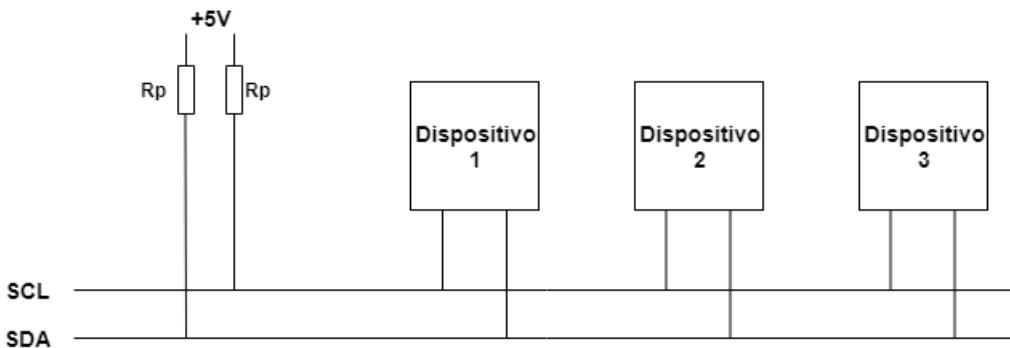


Figura 11 – Barramento I^2C .

O número de “nós” em um único barramento é limitado tanto pelo tamanho do endereço, que pode ser de 7 bits, 10 bits e até 16 bits; como por restrição de espaço, já que não se pode ultrapassar poucos metros de fios, pois a capacidade total máxima, algo em torno de 400 pF , impede o funcionamento correto do barramento.

Para que a informação seja enviada, o dispositivo mestre deve informar aos dispositivos escravos o início da comunicação onde o pino *SCL* deve estar em nível lógico alto e o pino *SDA* em nível lógico baixo. Quando isso ocorrer, todos os escravos estarão prontos para receber a primeira informação que é o endereço do escravo que comunicará com o mestre, junto com a operação que este escravo desempenhará. Em situações em que houver mais de um mestre na comunicação, terá preferência o mestre que sinalizar mais rápido o inicio de uma transmissão. Depois que o endereço é enviado, o escravo que tiver o endereço correspondente realizará a operação de leitura ou escrita da informação até que o dispositivo mestre informe para interromper a comunicação.

Como no barramento encontram-se vários dispositivos conectados, é necessário um resistor de *pull-up* em cada entrada (*SDA* e *SCL*). Calcula-se o valor mínimo ($R_p(min)$) e o valor máximo ($R_p(max)$) com as seguintes equações e os valores da tabela abaixo:

$$R_p(\min) = \frac{(V_{CC} - V_{OL}(\max))}{I_{OL}} \quad (3.11)$$

$$R_p(\max) = \frac{t_r}{(0.8473 \times C_b)} \quad (3.12)$$

	Parâmetros	Modo Padrão	Modo Rápido	Modo Rápido +	Unidade
t_r	Taxa de crescimento dos sinal SDA e SCL	1000	300	120	ns
C_b	Carregamento da capacidade para cada bus line	400	400	550	pF
V_{OL}	Baixo nível de tensão de saída (para 3_{mA} de corrente $V_{CC} > 2V$)	0.4	0.4	0.4	V
	Baixo nível de tensão de saída (para 2_{mA} de corrente $V_{CC} > 2V$)	-	$0.2 \times V_{cc}$	$0.2 \times V_{cc}$	V

Tabela 2 – Parâmetros retirados das especificações I^2C

Para a comunicação I^2C em *Fast-mode* com os parâmetros retirados das especificações da *Raspberry Pi 3* ($C_b = 200 \text{ pF}$ e $V_{CC} = 3.3 \text{ V}$) e da Tabela 2, calcula-se os valores de $R_p(\min)$ e $R_p(\max)$ de acordo com as equações (2.1) e (2.2):

$$R_p(\min) = \frac{(V_{CC} - V_{OL}(\max))}{I_{OL}} = \frac{(3.3 - 0.4)}{(3 \times 10^{-3})} = 966.667 \Omega \quad (3.13)$$

$$R_p(\max) = \frac{t_r}{(0.8473 \times C_b)} = \frac{(300 \times 10^{-9})}{(0.8473 \times 200 \times 10^{-12})} = 1.77 k\Omega \quad (3.14)$$

Com os resultados, adotou-se o valor de $1 \text{ k}\Omega$ para os resistores de *pull-up*.

3.1.3.1.2 Barramento *UART*

UART (*Universal Asynchronous Receiver/Transmitter*) é um modo de transmissão que transmite dados de um microprocessador para outro ou para um computador utilizando apenas dois fios (*RX/TX*). O *UART* é um sistema de comunicação *full-duplex*, ou seja, indica que o dispositivo pode transmitir e receber dados ao mesmo tempo, como pode ser visto na figura 12.

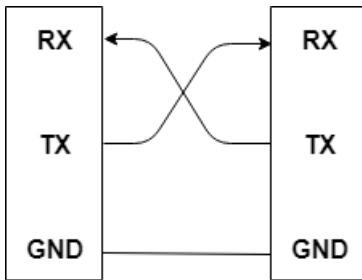


Figura 12 – Sistema de transmissão UART.

O pino de transmissão (*Tx*) do protocolo envia um pacote de bits que será interpretado bit a bit pelo pino receptor. Cada pacote enviado contém 1 *start bit* que indica o início da mensagem, 1 ou 2 *stop bits* para indicar o final da mensagem, 5 a 9 *bits* de informação e 1 *bit* de paridade para evitar a recepção de erros.

3.1.3.1.3 Barramento SPI

SPI (Serial Peripheral Interface) é um protocolo que permite a comunicação do microcontrolador com diversos outros componentes, formando uma rede. É uma especificação de interface de comunicação série síncrona usada para comunicação de curta distância, principalmente em sistemas embarcados.

Os dispositivos SPI comunicam entre si em modo *full duplex* usando uma arquitetura *master-slave* com um único mestre. O dispositivo mestre origina a protocolo para a leitura e a escrita. Múltiplos dispositivos escravos são suportados através de seleção com linhas de seleção de escravos individuais (CS ou SS).

O SPI pode ser descrito com precisão como uma interface de série síncrona, mas é diferente do protocolo síncrono de interface de série (SSI), que também é um síncrono protocolo de comunicação em série de quatro fios, mas emprega sinal diferencial e fornece apenas um único canal de comunicação simples.

Em modo *escravo*, o microcontrolador comporta-se como um componente da rede, recebendo o sinal de Clock. Em modo *mestre*, o microcontrolador gera um sinal de relógio e deve ter um pino de entrada/saída para habilitação de cada periférico, tal como é descritou pela Fig. 13.

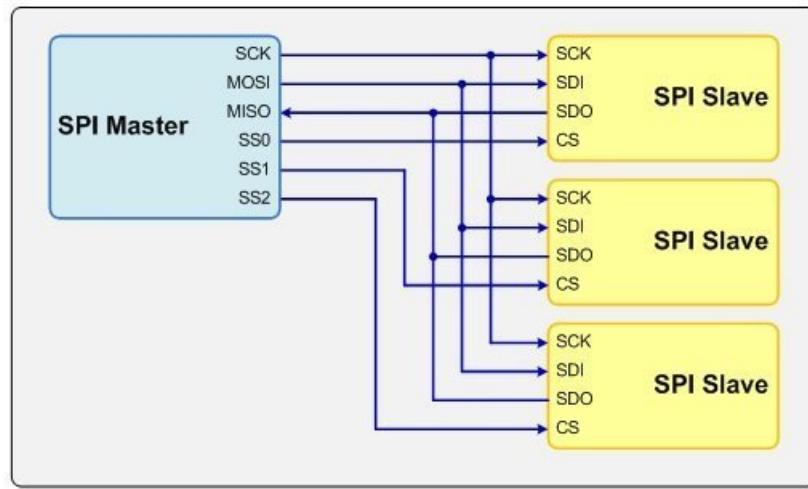


Figura 13 – Sistema de transmissão SPI.

Um dos pinos é denominado *MOSI*. A sigla *MOSI* significa Saída do Mestre e Entrada do Escravo (em inglês, *Master Output, Slave Input*). Como o próprio nome diz, a função dessa ligação é trafegar os dados que saíram do mestre e que devem chegar aos escravos.

Outra conexão é denominada de *MISO*, e é oposta ao *MOSI*. *MISO* significa Entrada do Mestre e Saída do Escravo (em inglês, *Master Input Slave Output*). A função dessa ligação é trafegar os dados que saíram do escravo e devem entrar no mestre. O terceiro fio que existe independente do número de escravos é o *SCK* (Clock). O *Clock* é um sinal que temporiza a comunicação e é controlado pelo dispositivo mestre. Esse sinal é responsável por fazer que mestre e escravo se comuniquem de forma sincronizada.

O pino *SS*, que significa Seleção de Escravo ou Seleção de *Chip* (do inglês, *Slave Select* ou *Chip Select*). Esse é um pino que controla com qual dispositivo escravo o mestre quer se comunicar. Cada escravo tem a sua entrada *SS* e só irá responder aos comandos se o nível lógico dessa entrada for 0 no momento da comunicação. Os *MISO* dos escravos que não foram selecionados via *SS* ficam em alta impedância, impedindo que a tensão que o escravo selecionado coloque em sua saída não danifique a saída dos outros escravos(CABRAL, 2015).

3.1.3.1.4 Módulo GPS

O Sistema de Posicionamento Global (em inglês *global positioning system - GPS*) é um sistema de radionavegação por satélite de propriedade do governo dos Estados Unidos e operado pela Força Aérea dos Estados Unidos. Trata-se de um sistema global de navegação por satélite que fornece informações de geolocalização e tempo a um receptor *GPS* em qualquer ponto da Terra, onde haja uma linha de visão desobstruída para quatro ou mais satélites *GPS*.

O *GPS* opera independentemente de qualquer recepção telefônica ou pela internet, embora essas tecnologias possam melhorar a utilidade das informações de posicionamento do módulo *GPS*. A tecnologia *GPS* fornece recursos de posicionamento críticos para usuários militares, civis e comerciais em todo o mundo, acessível gratuitamente a qualquer pessoa com um receptor *GPS*.

A utilização de um módulo *GPS* faz-se necessária no projeto para que tenha-se informações de posicionamento e velocidade do sistema. O módulo *GPS* escolhido para o projeto foi o *GY-NEO6MV2* que utiliza comunicação serial, permitindo, desta forma, integrar de forma eficaz uma comunicação com microcontroladores e placas como a *Raspberry Pi*. Além disso, trata-se de um módulo compacto e de alta confiabilidade nos dados fornecidos, tornando-se assim ideal para o projeto.

Entre as especificações do módulo *GPS GY-NEO6MV2*, ressaltam-se as seguintes:

- Alimentação: 2.7 V à 5 V (*DC*);
- Corrente de operação: 45 mA;
- Temperatura de operação: -40 à 85°C;
- Precisão: 5 m.

3.1.3.1.5 Módulo sensor *laser*

Um princípio muito utilizado a décadas em radares e sonares para determinar distâncias é o de medir o tempo que uma onda ecoada por um obstáculo leva para retornar a sua fonte emissora. No entanto, ondas de baixas frequências como as ondas sonoras e de rádio, sofrem difração ao se propagarem, tendendo a atingir obstáculos dispostos pelo caminho fazendo com que os sistemas de recepção se confundam.

Na física, a luz é tratada como partícula e o fóton sofre pouca difração. Os *lasers* produzem feixes de luz com alta direcionalidade que podem ser direcionados para obstáculos específicos refletindo um único feixe de luz para o equipamento de medição, viabilizando o seu uso para sistemas de medição de distância robustos e precisos.

Para a aplicação do projeto, o sensor a *laser* foi o escolhido para realizar a medição de distâncias devido a robustez e precisão dos sensores, o que não ocorre consequentemente com os sensores sonoros por causa da difração das ondas provocadas por diferença de temperatura, obstáculos e até mesmo a interferência do vento no meio externo, o que prejudica na precisão das medidas. Assim sendo, foi escolhido o sensor a *laser VL53L0X*.

O *VL53L0X* é um módulo de faixa de *laser* que fornece medições de distâncias exatas independentemente das refletâncias de alvo, ao contrário das tecnologias conven-

cionais, e pode medir distâncias absolutas de até 2 metros. O *VL53L0X* apresenta as seguintes especificações técnicas:

- Tensão operacional: 2.6 – 3.5 V;
- Temperatura de operação: –20 – 70°C;
- Emissor de infravermelho: 940 nm;
- Comunicação: I^2C .

O sensor laser ideal seria o HL-C2, porém o preço elevado deixou inviável essa escolha. Ele atenderia-nos devido às suas características otimizadas. As células receptoras de luz de alta densidade (em inglês *High Density Linear Cell* - HDLC) de tecnologia CMOS foram desenvolvidos especialmente para o sensor laser HL-C2 possuem uma grande velocidade de processamento que resultam em altas resoluções.

O Laser HL-C2 tem projeção de feixe gaussiano (*Micro Spot Gaussian Beam* - MSGB) em sua tecnologia de construção de abertura óptica. A lente de alta resolução reduz a abertura da lente o máximo possível e, desta forma, a luz entrando de qualquer ângulo pode ser reunida em um ponto mínimo para que obtenha-se uma maior precisão.

Devido ter diferentes distâncias do centro de medição e a lente do sensor do tipo ponto de feixe linear, as medições não são facilmente distorcidas por superfícies metálicas. Superfícies que parecem planas apresentam pequenas variações de superfície quando vistas sob ampliação. Essas variações podem causar erros na medição. Sensores do tipo ponto de feixe linear medem a influência dessas variações, permitindo a medição estável de peças de trabalho mais ou menos acabadas.

O sensor HL-C2 possui função de buffer que permite acumulação temporária no controlador de valores medidos a partir de amostragem de alta velocidade ($10s$), que são então transmitidos para o host, suportando no máximo 65.000 valores acumulados. O acúmulo de dados de forma pode contribuir para a rastreabilidade e outras atividades. Além disso, no modo de disparo os valores medidos antes e depois do erro podem ser adquiridos para ajudar a determinar a causa do erro, enviando uma entrada de acionador quando há um erro.

O HL-C2 suporta o protocolo MEWTOCOL (usado por controladores programáveis), o protocolo MC (usado pelas séries MELSEC-Q e MELSEC-L da Mitsubishi Electric) bem como o protocolo dedicado iQSS (usado pela MELSEC da Mitsubishi Electric -L series), permitindo que os valores medidos e outras informações sejam gravados automaticamente nos registros de dados dos controladores programáveis, sem necessidade de programação. As configurações de conexão HL-C2 podem ser feitas usando detecção automática de dispositivos conectados.

3.1.3.1.6 Módulo Acelerômetro/Giroscópio

Um acelerômetro trabalha utilizando a segunda lei de Newton, que define a força aplicada como sendo o produto da massa do corpo pela sua aceleração, ou seja, ao medir a aceleração aplicada sobre um corpo é possível determinar a força aplicada sobre ele. Assim sendo, um acelerômetro consiste em um instrumento capaz de medir a aceleração sobre objetos.

Ao escolher um acelerômetro deve-se levar em conta parâmetros como:

- Amplitude de vibração: É importante que a amplitude de vibração não seja superior à especificada pelo fabricante, pois se assim for, pode distorcer ou atenuar o sinal de saída;
- Sensibilidade: É a tensão de saída produzida por determinada força medida em g 's. A sensibilidade pode ser afetada pela frequência de vibração, pois o dispositivo é apto a operar dentro de um intervalo de frequência fornecido pelo fabricante;
- Frequência de vibração: Os sensores de aceleração possuem intervalos de frequência que podem trabalhar. Dessa forma a frequência deve estar contida na faixa especificada pelo fabricante, pois caso seja menor não há uma detecção correta e o contrário pode saturar o sinal de saída ou até mesmo danificar o sensor;
- Número de eixos: Acelerômetros triaxiais medem as vibrações em três eixos, X , Y e Z . Eles possuem três cristais, posicionados de modo que cada um reaja à vibração em um eixo diferente, produzindo um sinal específico para cada eixo.

O giroscópio é um dispositivo onde o eixo de rotação mantém sempre a mesma direção na ausência de forças que o perturbem. Seu funcionamento baseia-se no princípio da inércia. O giroscópio não possui uma funcionalidade radicalmente diferente do acelerômetro, ele é um sensor que utiliza da força da gravidade para indicar a posição de um determinado objeto no espaço, podendo identificar se você gira algo em seu próprio eixo ou saber se esta apontando ele para cima ou para baixo.

Com base na discussão acima, foi escolhido o *MPU-6050*, pois apresenta características satisfatórias que atendem aos requisitos do projeto, além de possuir tanto o acelerômetro quanto o giroscópio em um único módulo. Dentre as especificações do *MPU-6050*, se destacam as seguintes características:

- Protocolo de comunicação: I^2C ;
- Três conversores analógico-digital (AD) de 16 bits para digitalizar as saídas do giroscópio e mais três para o acelerômetro;

- Alimentação: $2.375\text{ V} - 3.46\text{ V}$;
- Sensibilidade do giroscópio: $\pm 250, \pm 500, \pm 1000$ e $\pm 2000\text{ }^{\circ}/\text{s}$;
- Sensibilidade do acelerômetro: $\pm 2, \pm 4, \pm 8$ e $\pm 16\text{ g}$.

3.1.3.2 Arquitetura da Placa de Circuito Impresso

Na confecção de uma Placa de Circuito Impresso, ou da sigla PCB em inglês, existem boas práticas que devem ser sempre aplicadas. Os objetivos principais de projeto da arquitetura de uma PCB visam minimizar problemas com compatibilidade eletromagnética (EMC), loops de Terra, acoplamentos, entre outros problemas comuns em PCB.

Um Plano de Terra de baixa indutância é fundamental para minimizar problemas de EMC. Maximizar as áreas de terra de uma PCB reduz a indutância de terra em um sistema, que por sua vez reduz as emissões eletromagnéticas. Não é aconselhável conectar todos os terras individuais e, em seguida, conectá-los ao plano de terra, pois aumenta o tamanho da malha de corrente. Os sinais podem acabar se acoplando ao terra de diferentes maneiras. Caso os componentes sejam ligados de maneira aleatória ao terra do sistema, teremos uma PCB de baixa qualidade. Esse problema de *layout* gera alta indutância e problemas sérios de EMC.

A forma como um sinal de volta à terra do sistema é muito importante, pois quando um sinal toma um caminho mais longo, ele cria um loop de terra, que forma uma antena e irradia energia. Assim, toda a trilha que flui corrente de volta para a fonte deve seguir o caminho mais curto possível, e deve ir diretamente para o plano de terra.

Sabendo disso, projetamos um plano inteiro de terra, pois isso proporciona menor impedância para as correntes de retorno.

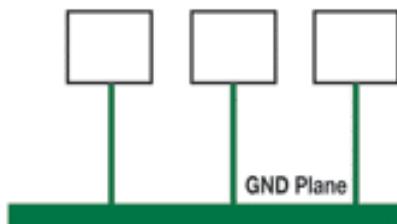


Figura 14 – *Layout* recomendado do GND

A maioria dos problemas relacionados com EMC são causadas por cabos que transportam sinais digitais que atuam efetivamente como uma antena eficiente. Idealmente, a corrente que entra um cabo sai do outro lado, mas na realidade as coisas são diferentes por causa da indutância e capacidade parasitas que emitem radiação. A solução de pre-

venção será de usar um cabo de par trançado para ajudar a manter o acoplamento a um nível baixo, minimizando os campos magnéticos induzidos.

Os componentes na placa projetada serão agrupados de acordo com sua funcionalidade, tais como, seções analógicas, seções digitais, fonte de alimentação, entre outros. As trilhas para cada grupo estarão em sua área designada.

Além da separação em seções, haverá idealmente a separação das alimentações por planos de funções, fazendo a ligação entre os planos de terra em pontos definidos.

3.2 Energia

O objetivo consiste na alimentação de todo o sistema elétrico necessário para a aquisição de dados e análise, sendo esta alimentação feita de forma autônoma em relação ao veículo ao qual o protótipo será acoplado.

3.2.1 Gerador de energia

Como dito anteriormente, foi adquirido um motor semelhante ao utilizado em sistemas de geração eólica (figura 13). No projeto, este será utilizado como um dínamo, a fim de gerar a energia através da rotação de seu eixo, sendo esta transmitida pelo contato com uma das rodas. De acordo com as especificações do motor, para garantir a potência desejada (35w) ele deve operar entre 1900 e 2500 rpm, para isso devemos calcular a relação entre a polia e o pneu para que alcance a velocidade desejada quando o carro estiver a velocidade de 30km/h.



Figura 15 – Gerador de energia

3.2.1.1 Dínamo

O alternador e o dínamo, são equipamentos que utilizam o princípio da indução eletromagnética.

A geração de um campo magnético indutor pode ser efetuada através de ímãs permanentes ou de eletroímãs. Independentemente disso, os geradores de energia elétrica, podem ter dois tipos construtivos: O indutor é o estator e o induzido é o rotor, o indutor é o rotor e o induzido é o estator. O dínamo só pode ser feito do primeiro modo, isto é o indutor é o estator e o induzido é o rotor. A f.e.m. induzida no rotor tem de ser recolhida para o exterior através de escovas em contato com os segmentos do coletor.

A princípio, a fim de obter uma melhor geração, pensou-se em um alternador acoplado a uma polia e assim iria ocorrer a transferência do movimento das rodas para o eixo do alternador, contudo, foram realizados testes em bancada e foi constatado que estava sendo gerado uma baixa quantidade de energia com a rotação que podemos obter com as rodas do protótipo. O trabalho até o presente momento enfrenta dificuldades em relação a movimentação do protótipo pois grandes velocidades acarretam para o projeto como um todo, grandes vibrações e perdas na aquisição de sinais da parte eletrônica.

Com isso, uma segunda alternativa encontrada foi a geração de energia por um circuito formado por um dínamo, um retificador de tensão e uma bateria. Para que haja transmissão de movimento, foi acordado com a gestão de estrutura do projeto uma , a fim que não haja problemas para nenhuma das vertentes.

Apesar de ainda não ter havido uma integração com a estrutura do perfilômetro, foi construído, a fim de simular a funcionalidade do subsistema energético, um pequeno protótipo com uma roda aro 13, a qual transmitirá o movimento para o eixo do dínamo por meio de uma segunda roda de tamanho menor, como já exposto na figura 14.

Características:

- Tipo: Corrente contínua com escovas;
- Tensão: 12V a 2000 rpms
- Engrenagem: 18 dentes
- Medidas: 180mm x 48mm de diâmetro, Cabo 24cm
- Corrente: 5A em 12v
- Peso: 1,2kg

3.2.1.2 Polia

Para que seja definido o tamanho da polia, primeiro é preciso saber a potência necessária e por sua vez a velocidade para que se gere tal potência. De acordo com os componentes utilizados chegamos a uma demanda de aproximadamente 35W. Para determinar a rotação do gerador foi efetuado um teste medindo a velocidade em relação a potência gerada, como podemos ver na tabela abaixo.

RPM	Tensão (V)	Corrente (A)	Potência (W)
3100	14	4.5	63
2340	12.4	3.5	43.4
1700	11	2.5	27.5

Tabela 3 – Teste de Velocidade x Potência

Dessa forma podemos fazer uma interpolação e chegar ao valor de 2000 RPM para 35W e com isso calcular a relação entre as polias da seguinte forma:

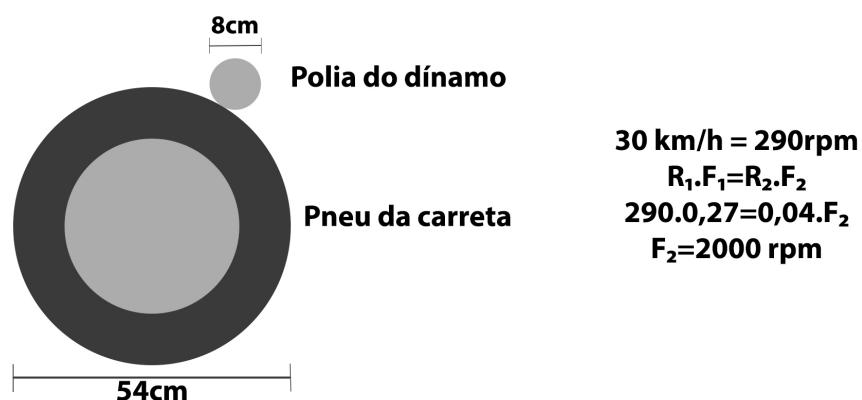


Figura 16 – Relação a polia e o pneu

3.2.2 Recarga da bateria

O motor gera uma corrente contínua que deve ser estabilizada e levada para o carregamento da bateria. Dessa forma, foi feito um circuito que estabiliza esta tensão de entrada, reduz a corrente de entrada e promove a carga da bateria.

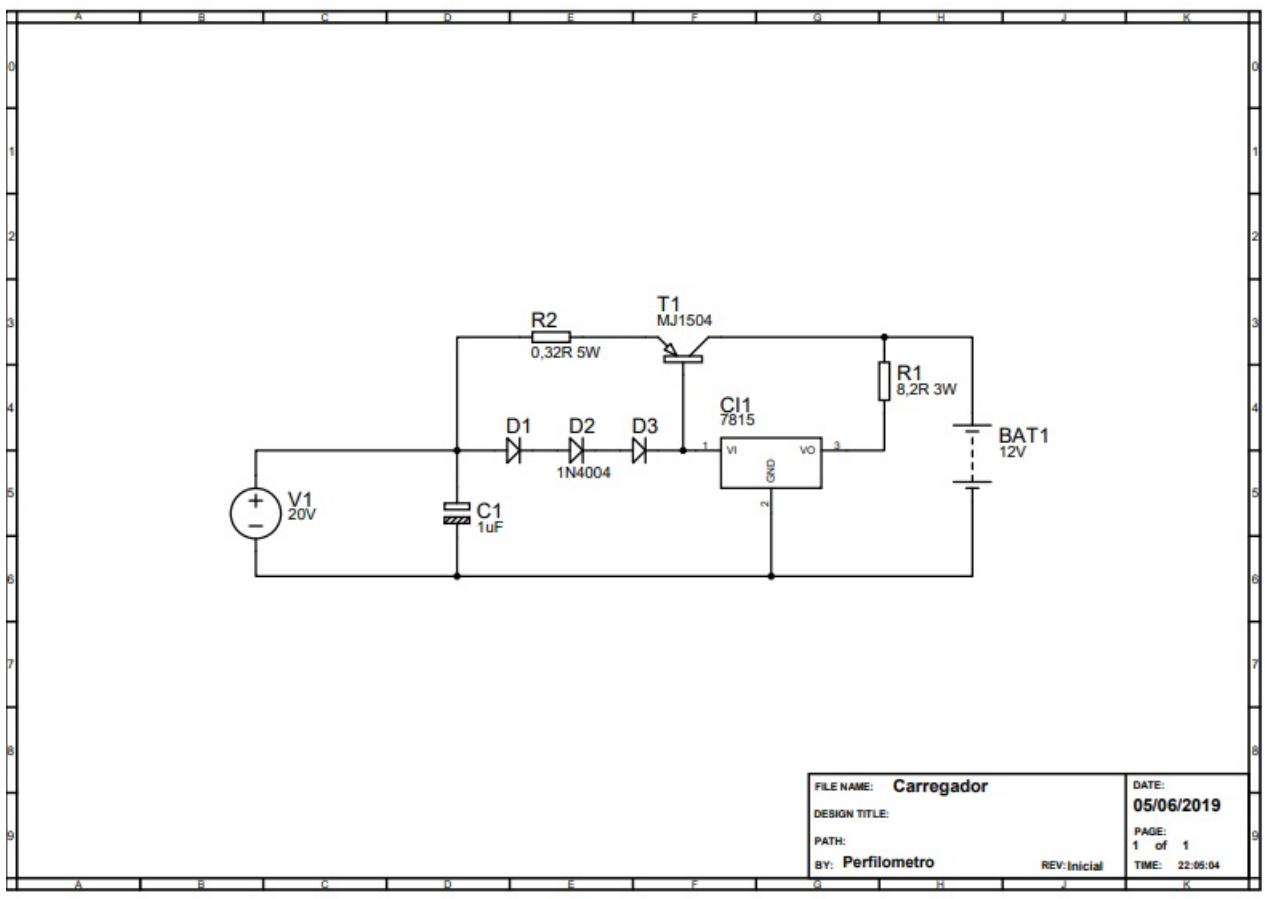


Figura 17 – Circuito do carregador de bateria

3.2.2.1 Bateria 12V 7Ah

A bateria foi escolhida de acordo com os requisitos da equipe de eletrônica, o controlador necessita que seja uma bateria de 12v e a parte eletrônica que tenha uma corrente de descarga de até 3A, a bateria de 7Ah é a capaz de fornecer a potência necessária com uma folga razoável. Além disso é necessário que a bateria tenha a tecnologia VRLA, ou seja, tem baixa resistência interna para maior corrente instantânea e maior rapidez na recarga. A válvula VRLA controla pressão interna dos gases, garantindo que não emita gases corrosivos e que possa ser instalada próxima a circuitos elétricos.

3.2.2.2 Regulador de tensão DC-DC

O subsistema eletrônico necessitará de uma alimentação inferior ao que é fornecido pela bateria. A fim de solucionar o problema, foi adquirido um módulo Regulador de Voltagem DC-DC 5A com Voltímetro, comercialmente denominado STEP DOWN.

O módulo utilizado possui 180 KHz de frequência fixa PWM para redução de voltagem (step-down) módulo DC / DC, é capaz de conduzir uma carga 5A com alta eficiência, baixa ondulação e excelente linha e regulação de carga. Possui um medidor

de tensão para exibir a tensão de voltagem de entrada e de saída, e a tensão pode ser corrigida e melhorar a precisão do voltímetro.

A escolha deste regulador, deveu-se ao fato das diversas ligações que serão feitas ao subsistema de eletronica, necessitarem de estabilidade na tensão e várias faixas de corrente.

Características:

- Voltagem de Entrada: 4 - 38VDC
- Voltagem de Saída: 1,5 - 36VDC
- Corrente de saída Max: 5A
- Potencia de Saída: 75W
- Variação do Voltímetro: 4 a 40V, precisão 0,2V
- Dimensões: 6,6cm x 3,9cm x 1,8cm

3.3 Estrutura

O objetivo consiste na análise estrutural de um reboque leve utilizado para o acoplamento do perfilômetro. Deve-se fazer um estudo do chassi e suas reações de apoio, assim como o grau de liberdade de cada parte para análise das tensões atuantes e validação do projeto. Foi utilizado simulações numéricas MFE em conformidade com as legislações vigentes para o cálculo de tensões e deformações do chassi e componentes, visando assegurar sua integridade durante e após os esforços estáticos serem aplicados.

3.3.1 Estrutura do Reboque

De acordo com os princípios definidos pelos órgãos fiscalizadores das propriedades funcionais de segurança e aplicabilidade dos reboques em vias urbanas, é preciso delimitar os parâmetros de projeto em função das leis definidas pelo o CONTRAN (conselho nacional de trânsito) especificado o equipamento em relação aos objetivos em que o mesmo tende a desempenhar. Os reboques por definição são definidos em semi reboques (são apoiados nas unidades tratoras ou interligados a ela por meio de articulações) e reboques (engatados na parte posterior de veículos tratores), estes equipamentos podem ser fabricados artesanalmente de acordo com o artigo 106 da norma elaborada pelo o CONTRAN que verbaliza que caso o fabricante altere ou substitua áreas específicas da estrutura as alterações devem ser avaliadas por órgãos metrológicos específicos. O processo de fabricação é avaliado em função do peso bruto do reboque que é definido pela soma do seu peso estrutural e sua carga útil, definido abaixo:

$$\text{PBT} = \text{Estrutura} + \text{Carga útil}$$

De acordo com o valor do PBT, os reboque podem ser separados em leves ou pesados. Para estruturas com peso bruto abaixo de 500 kgf o reboque é definido como leve , entretanto, se o veículo possuir valor de peso bruto acima de 500 kgf o reboque é tido como pesado, onde exigem um maior cuidado com as normas de seguranças como por exemplo um sistema de freio que abranja o freio inercial e o de estacionamento. Para fins deste projeto as solicitações do PBT não excederão 500 kgf, portanto o reboque escolhido será atentado as características do reboque leve.

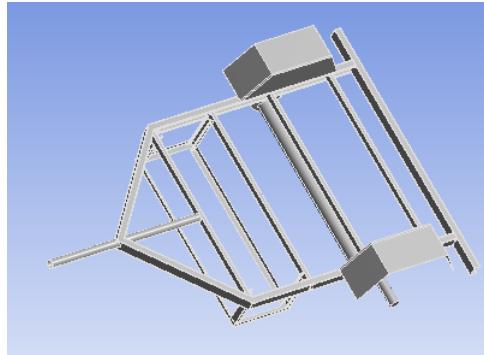


Figura 18 – Reboque utilizado

Os equipamentos de uso obrigatório são definidos abaixo pelas normas do OIC:

- Sinalização
- Iluminação
- Direção
- Eixo e suspensão
- Pneus e rodas
- Sistemas de componentes regulamentares

De acordo com o Regulamento Técnico de Qualidade 25 (RQT 25), a força de reação no engate não deve exceder **700 N**.

3.3.1.1 Desenho Técnico

As vistas do reboque utilizado são dadas abaixo:

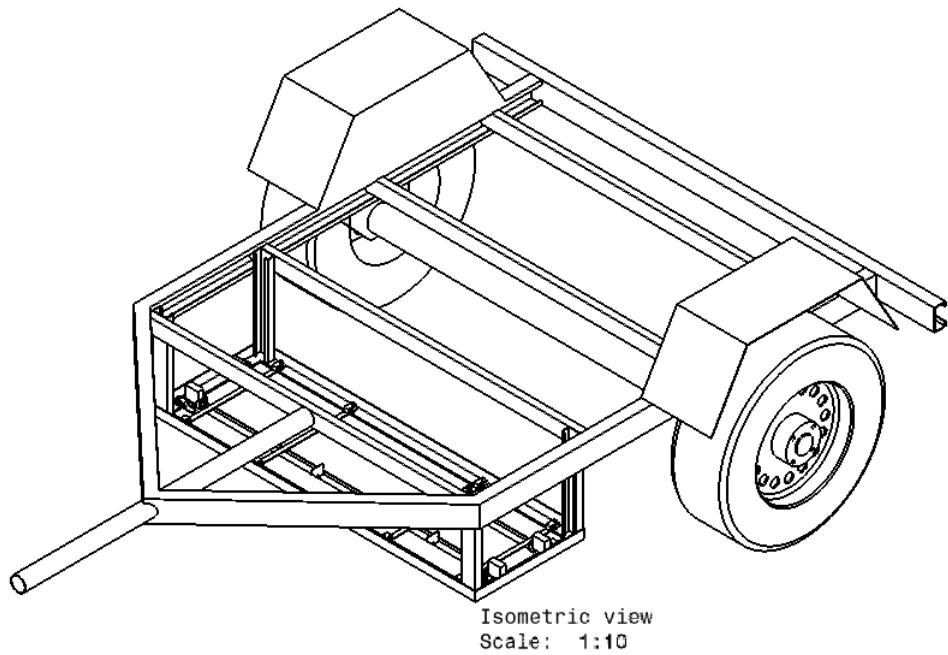


Figura 19 – Vista Isométrica

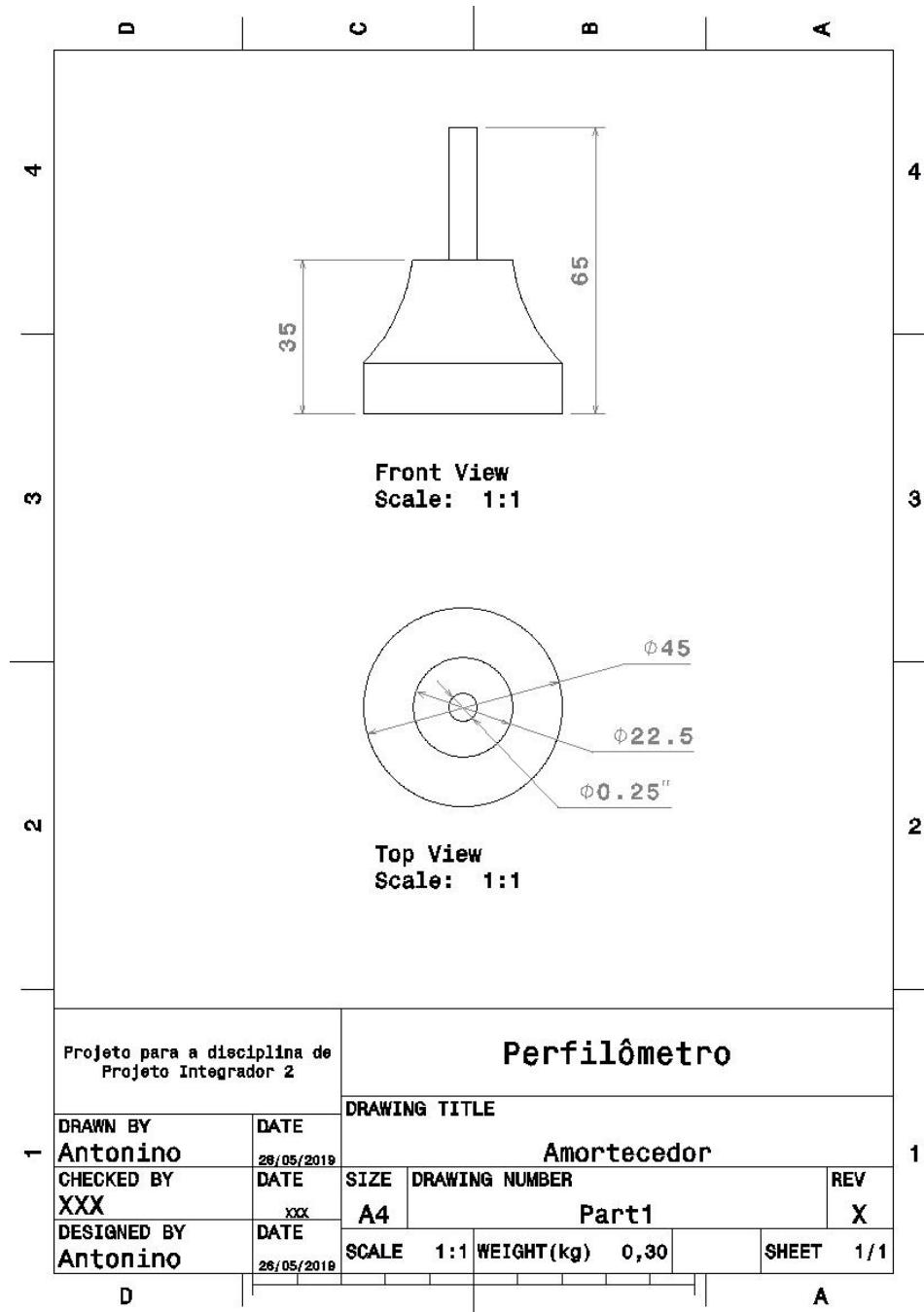


Figura 20 – Amortecedor

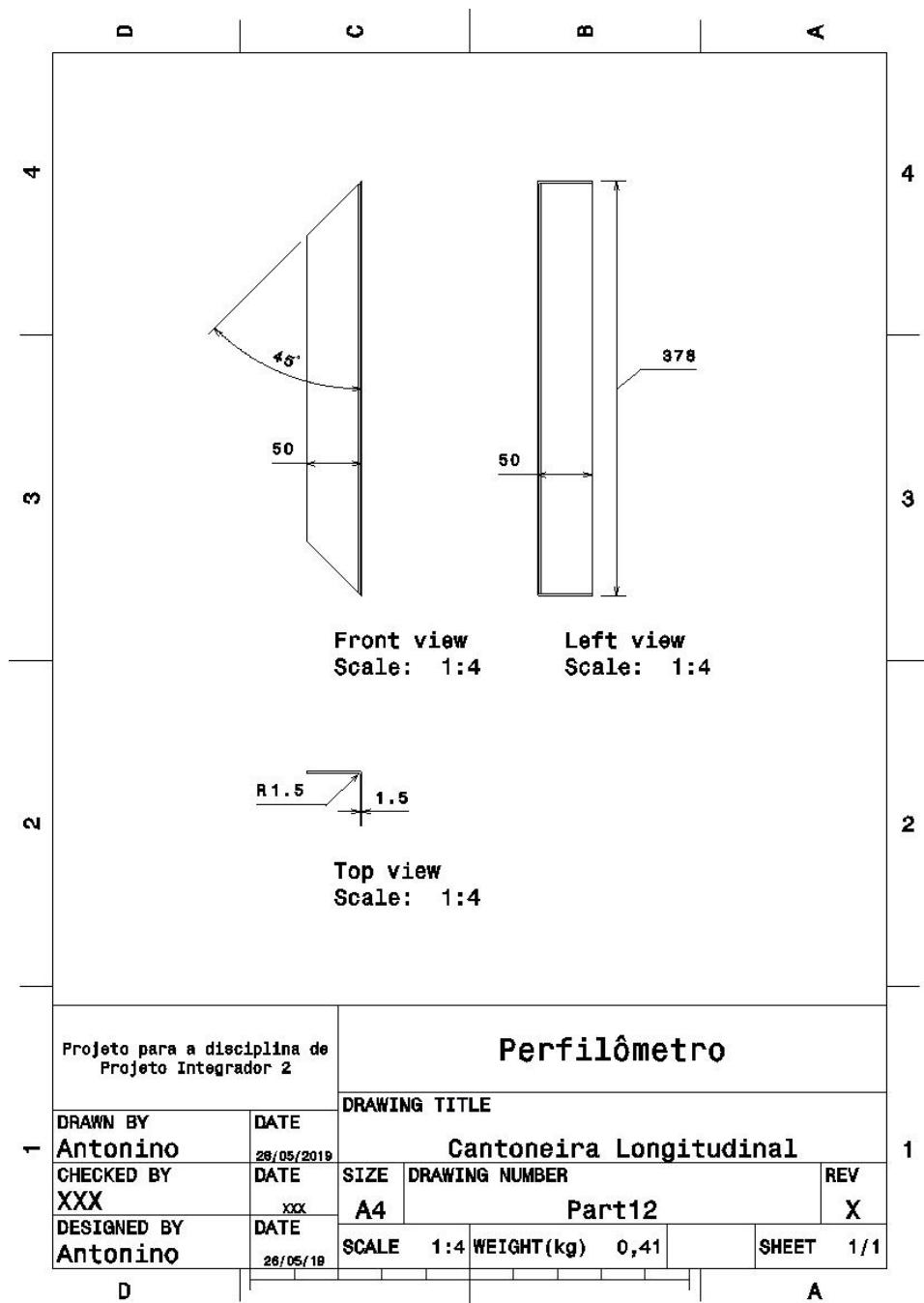


Figura 21 – Cantoneira longitudinal

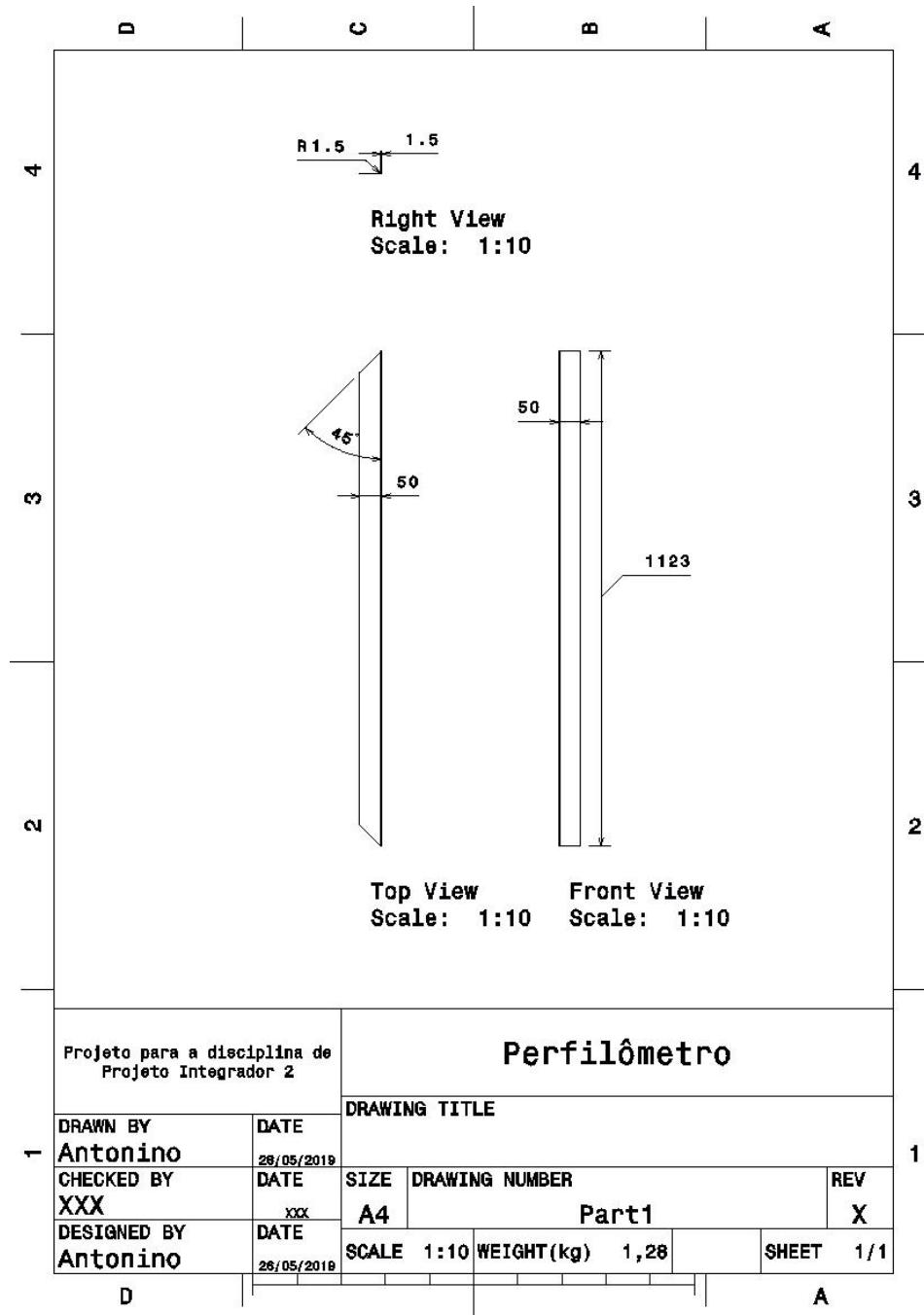


Figura 22 – Cantoneira transversal

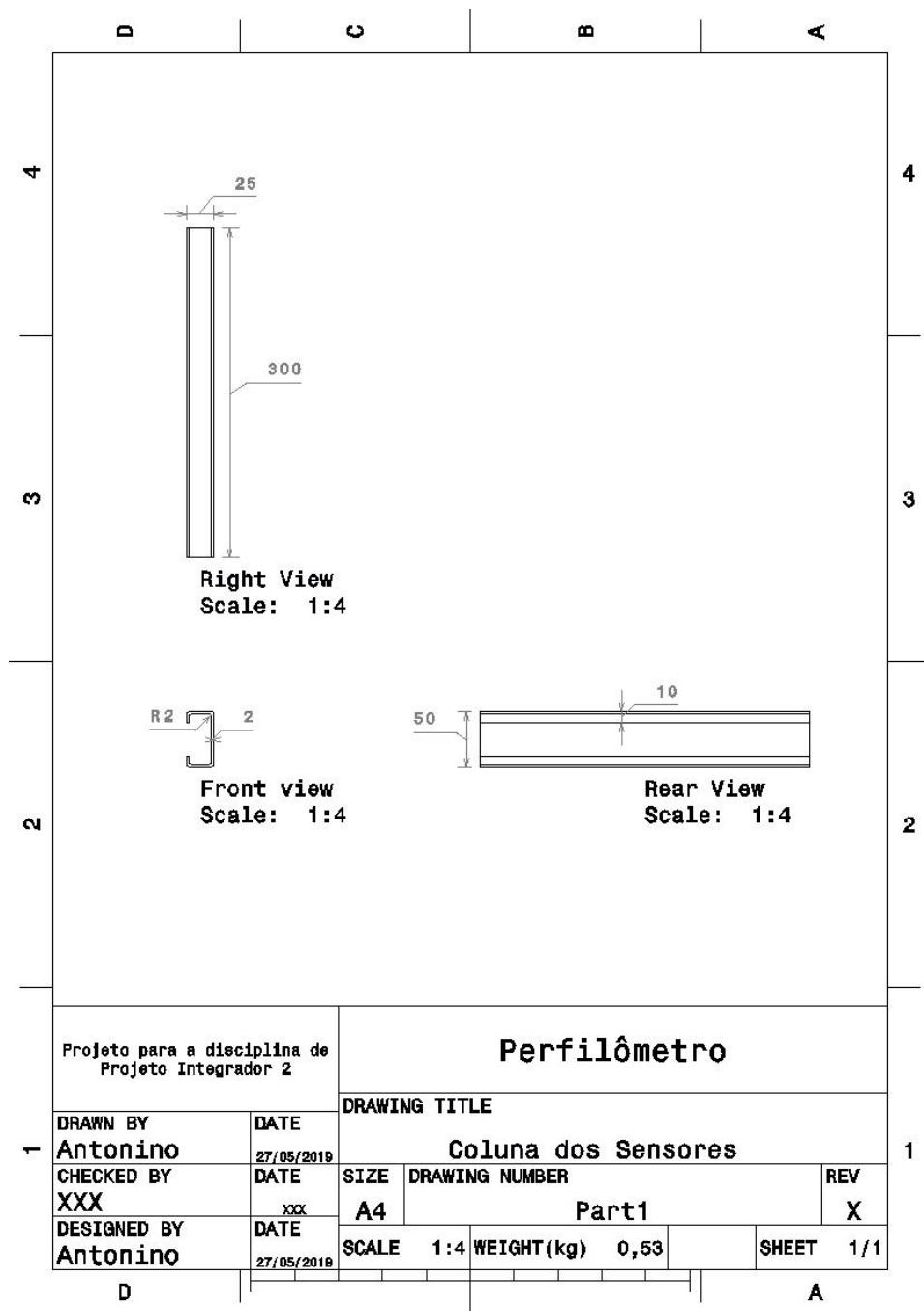


Figura 23 – Coluna frontal

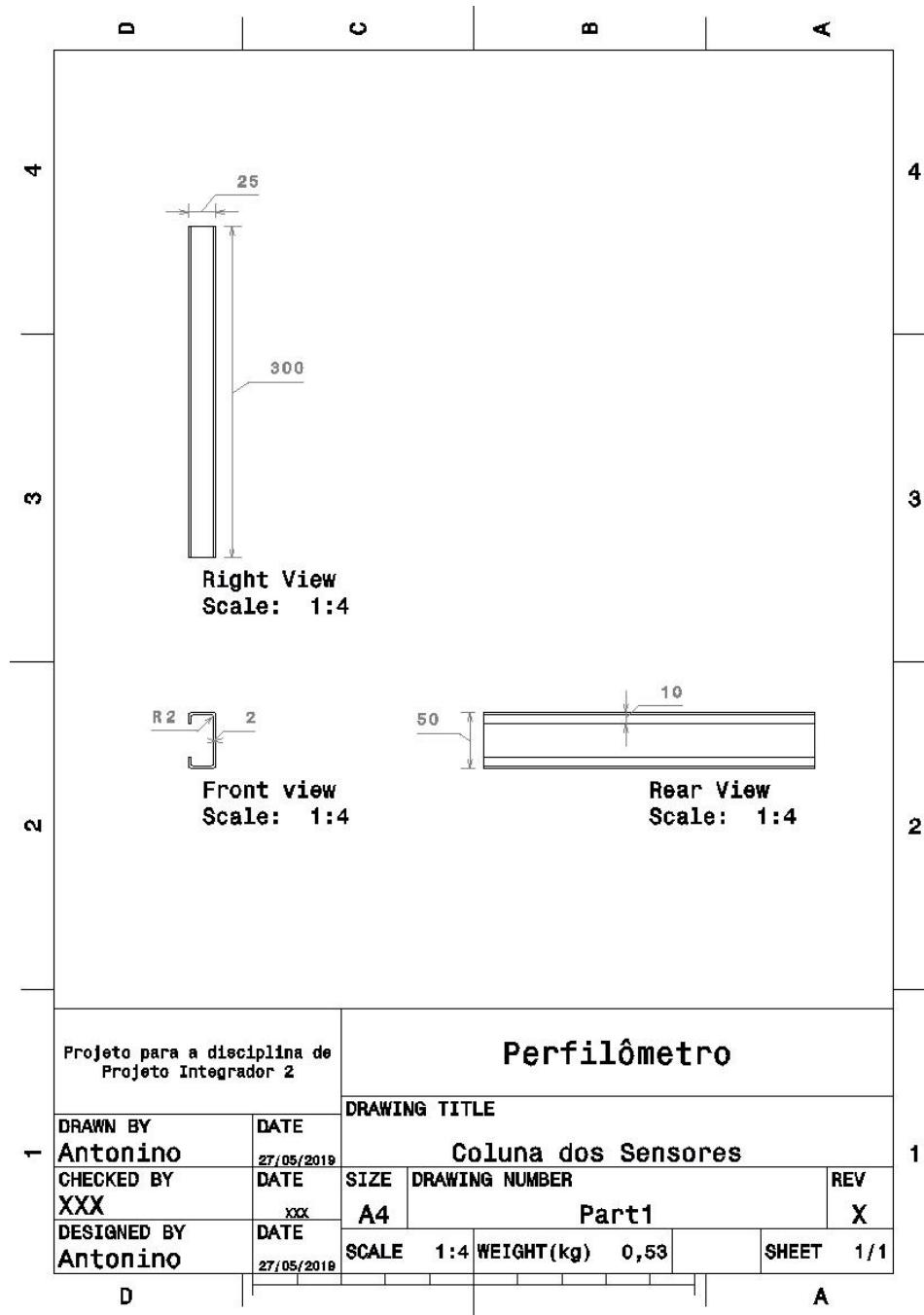


Figura 24 – Coluna frontal

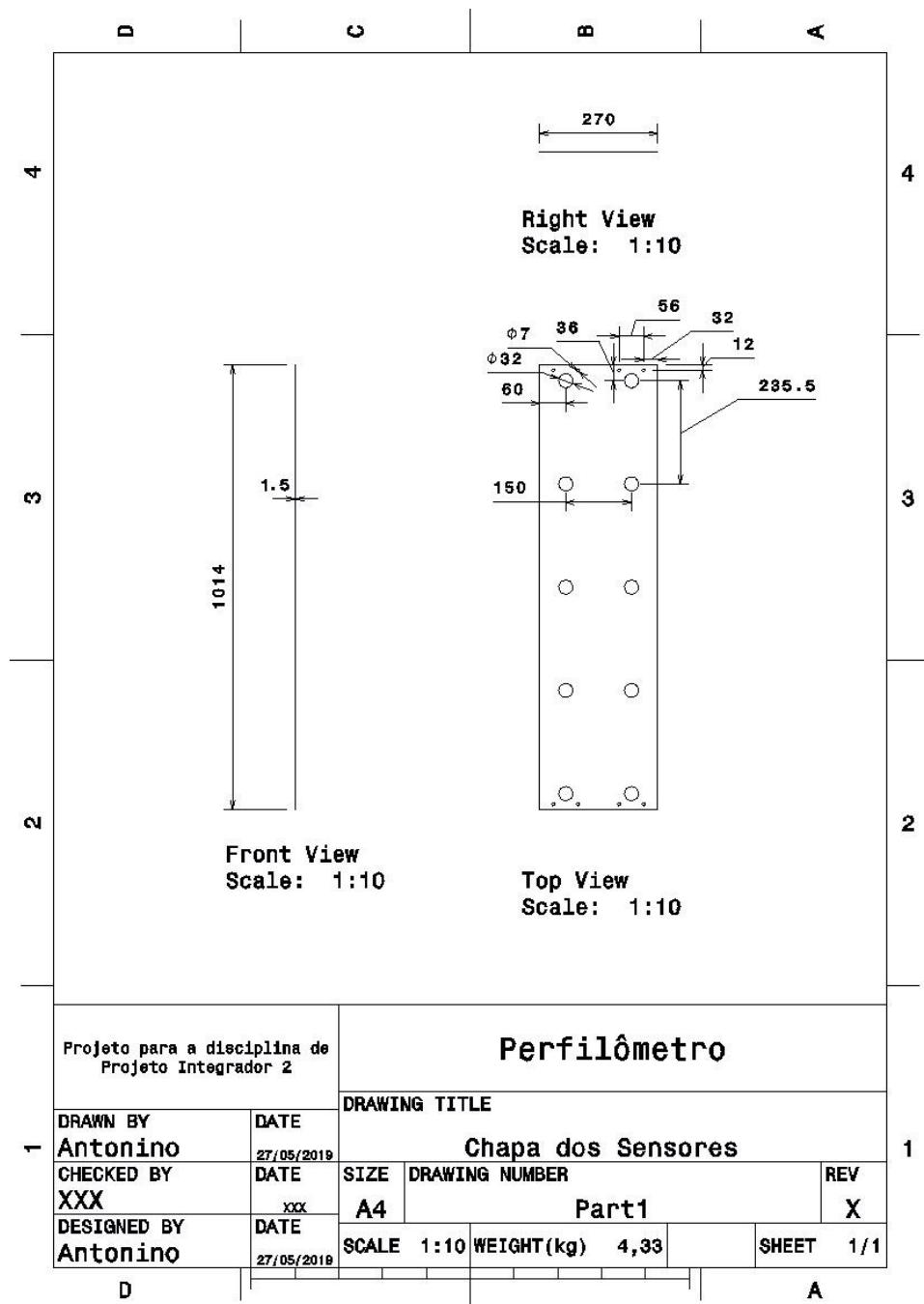


Figura 25 – Chapa dos sensores

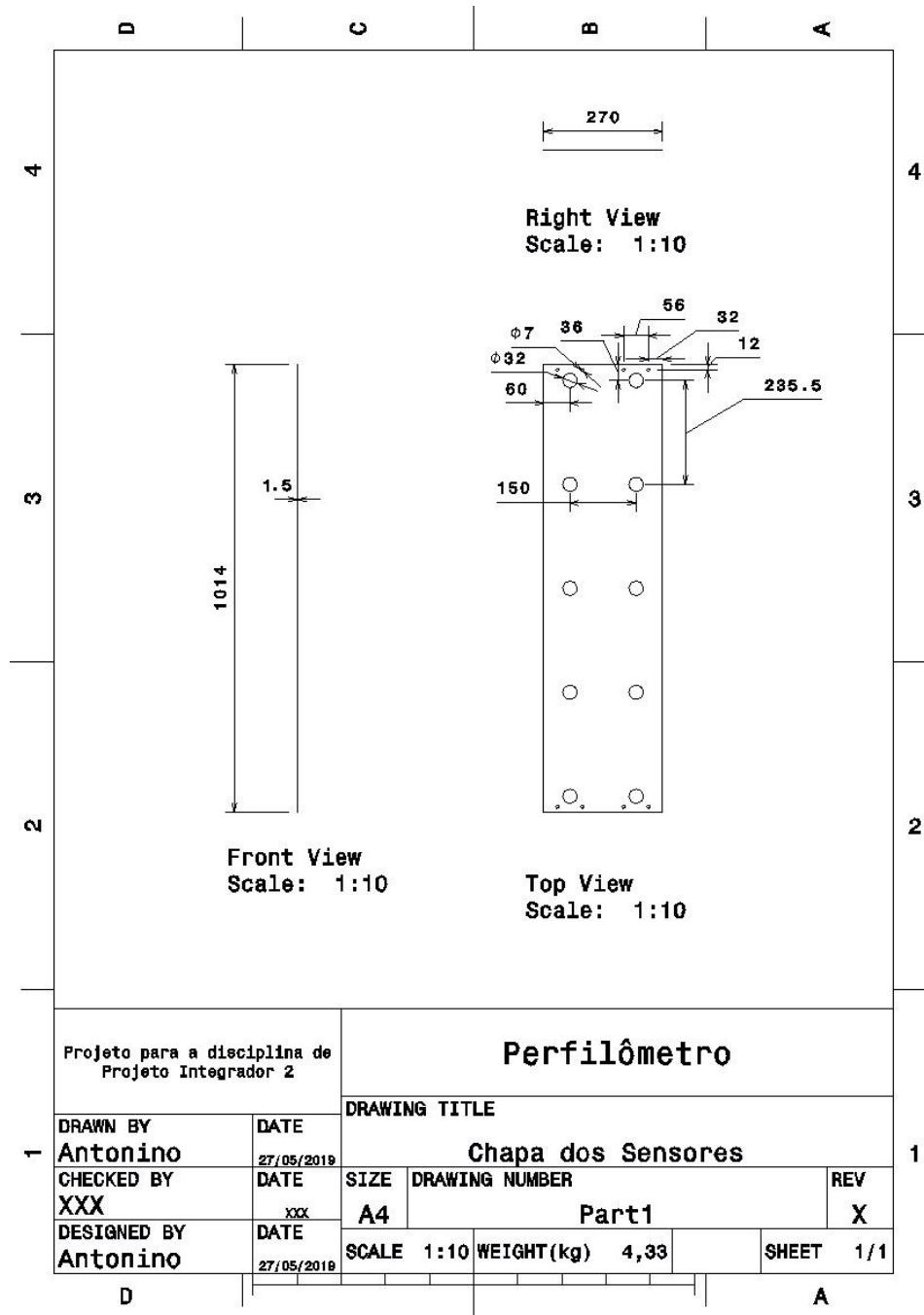


Figura 26 – Chapa dos sensores

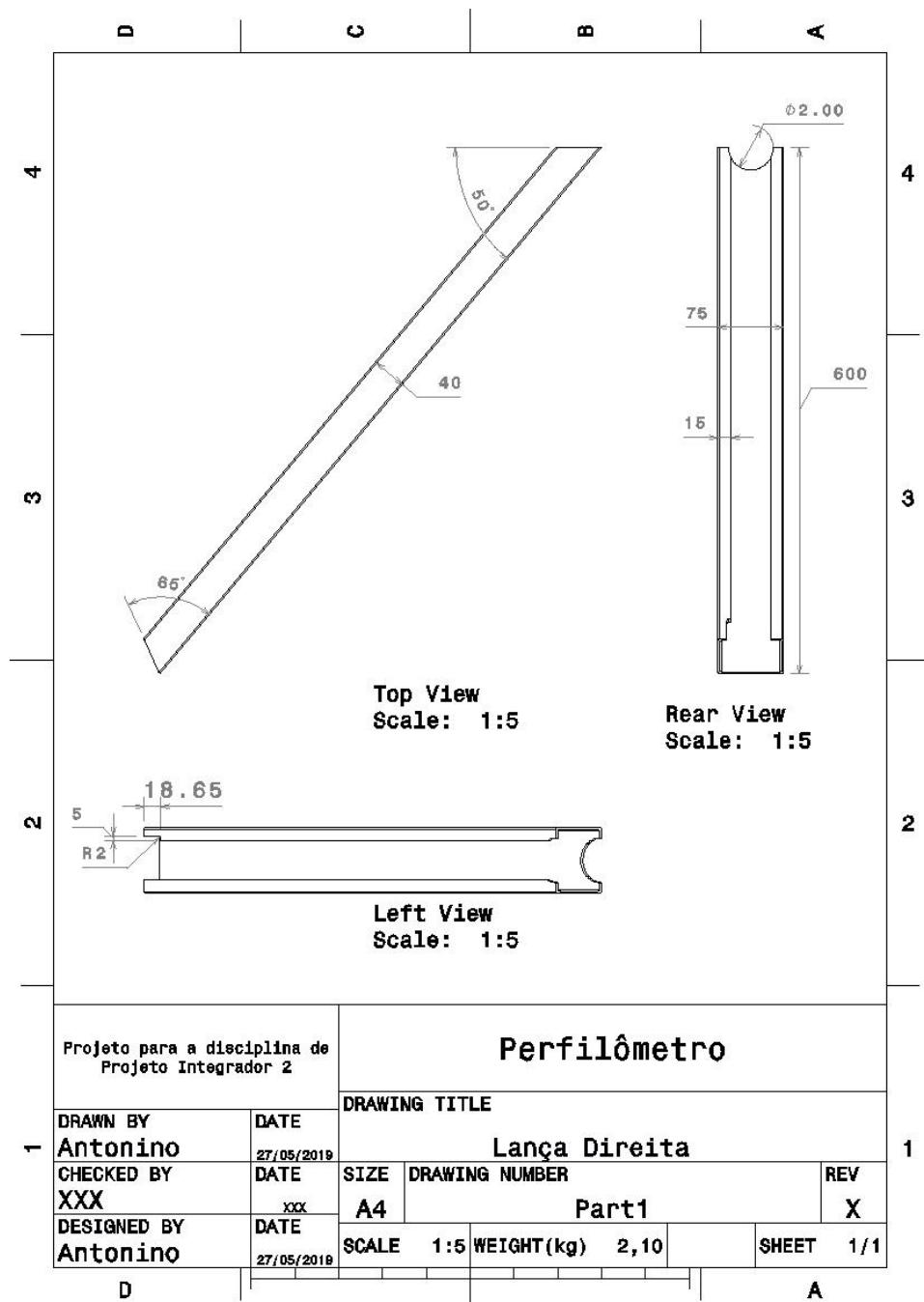


Figura 27 – Lança direita

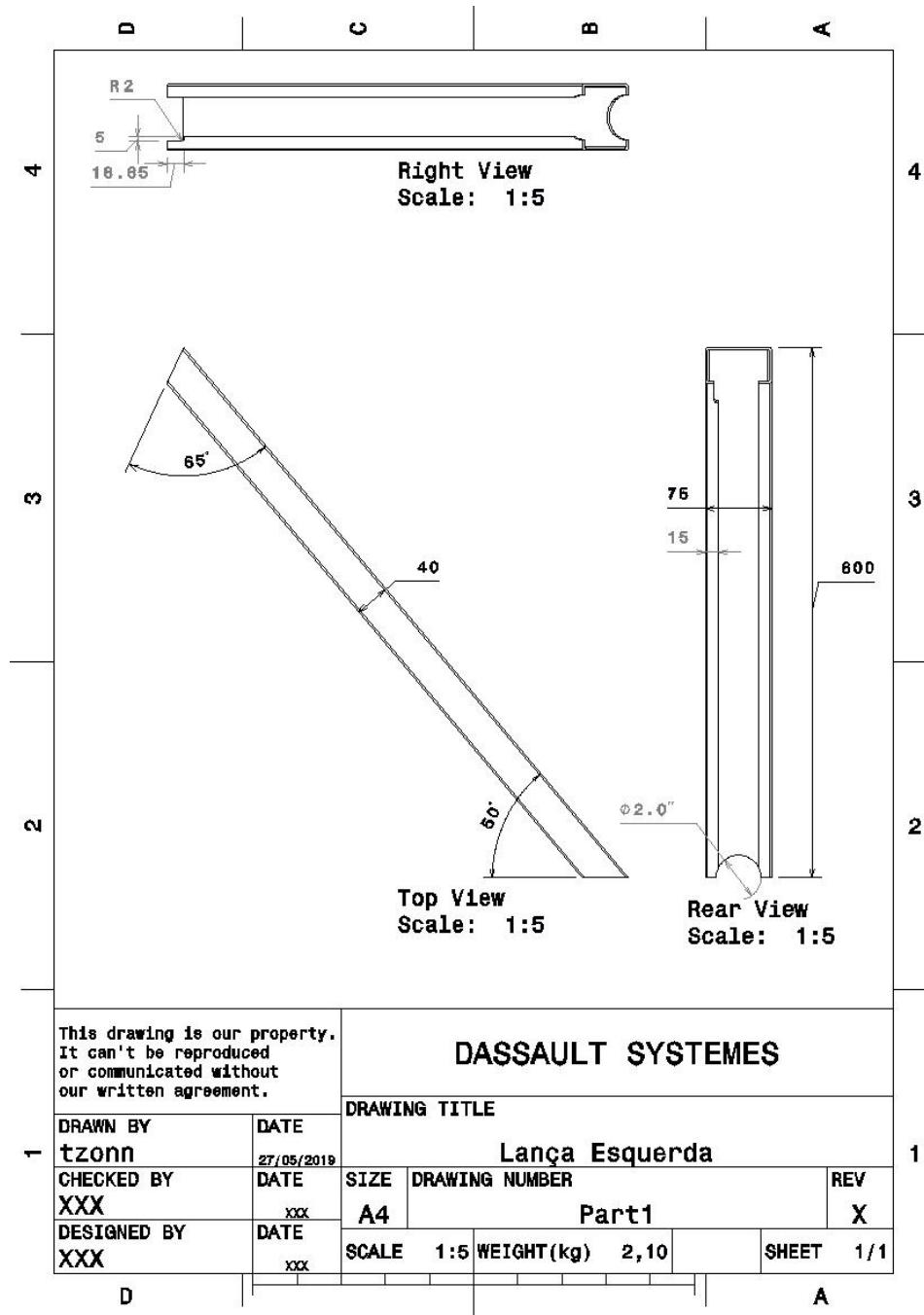


Figura 28 – Lança esquerda

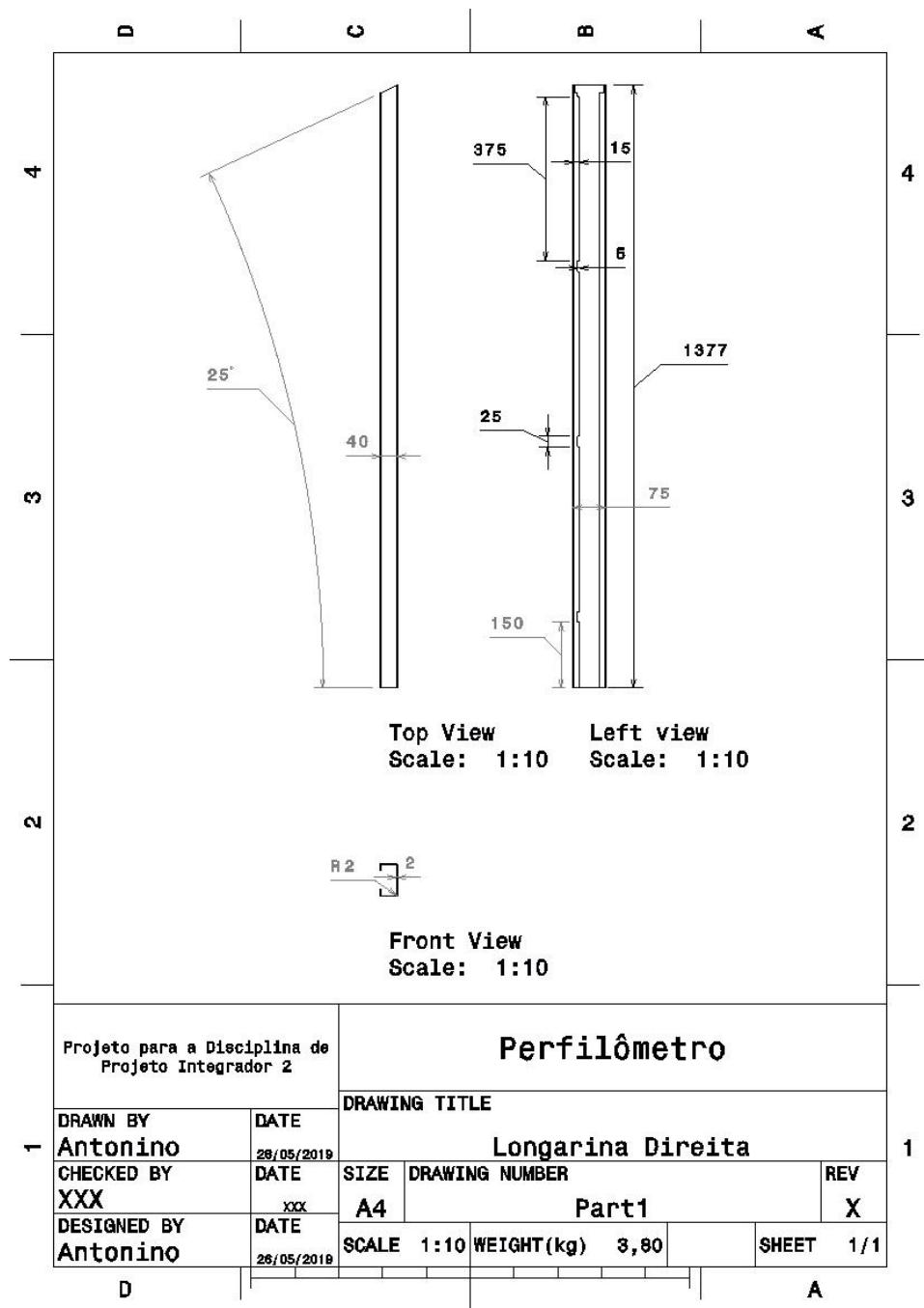


Figura 29 – Longarina direita

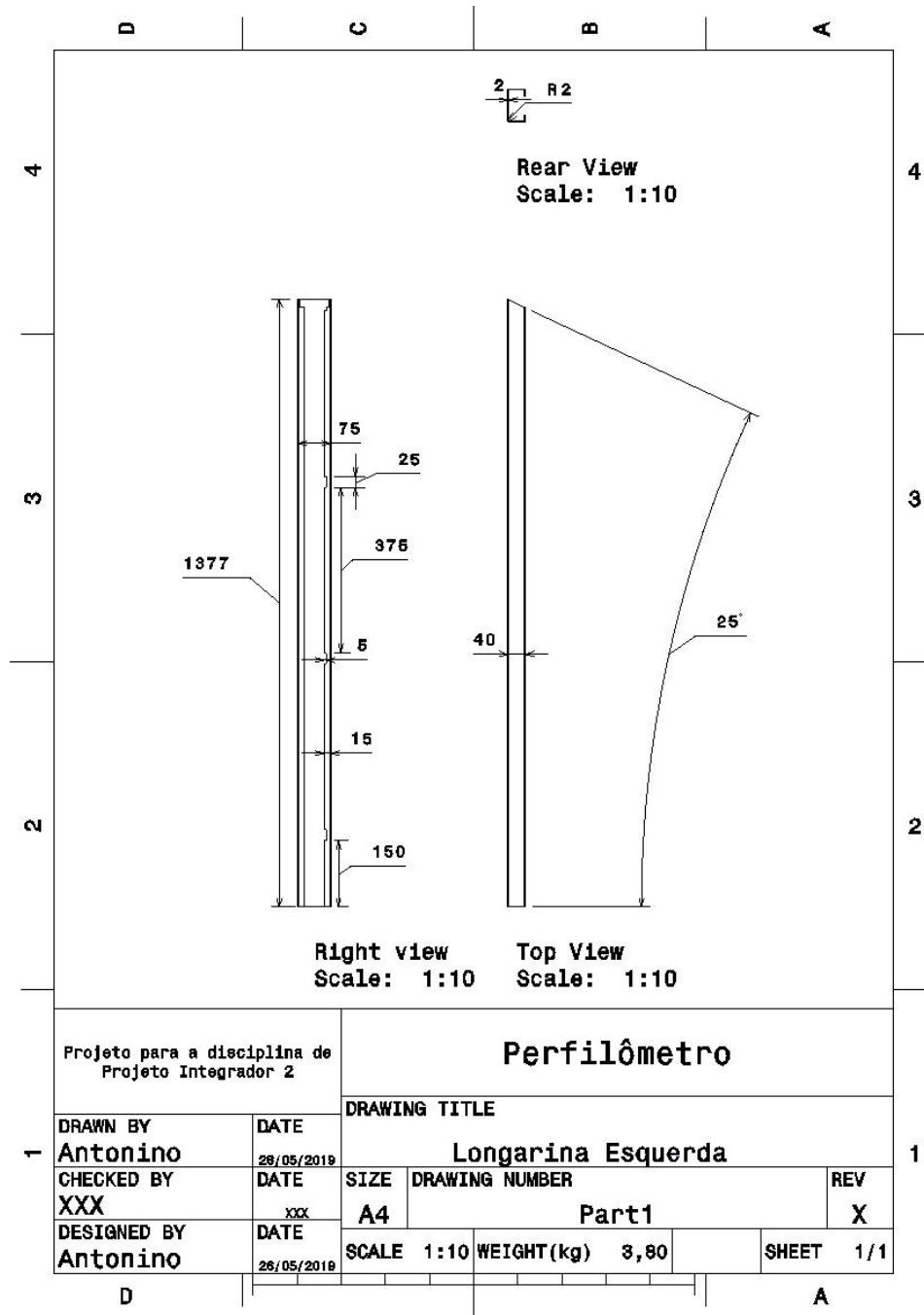


Figura 30 – Longarina esquerda

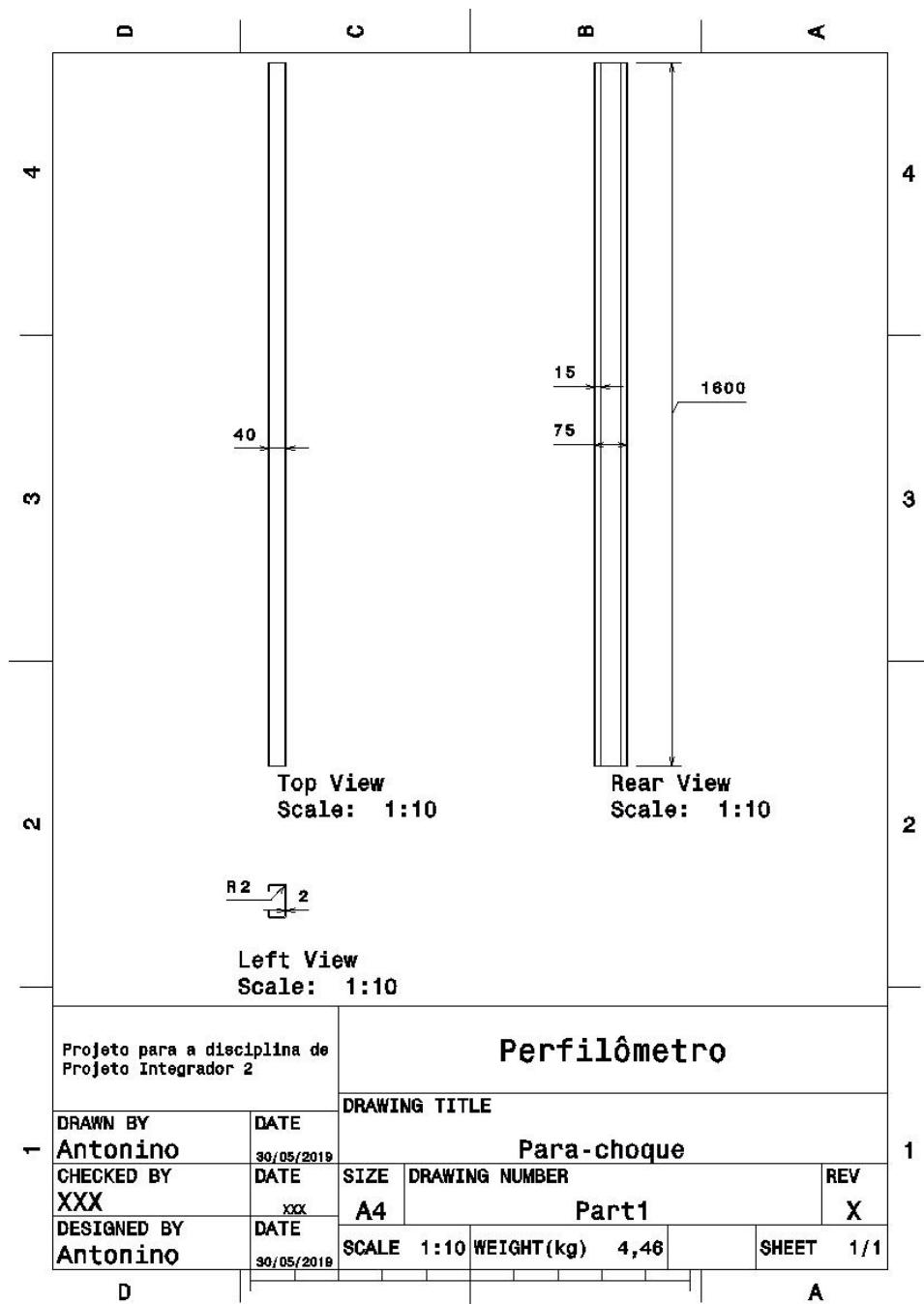


Figura 31 – Parachoque

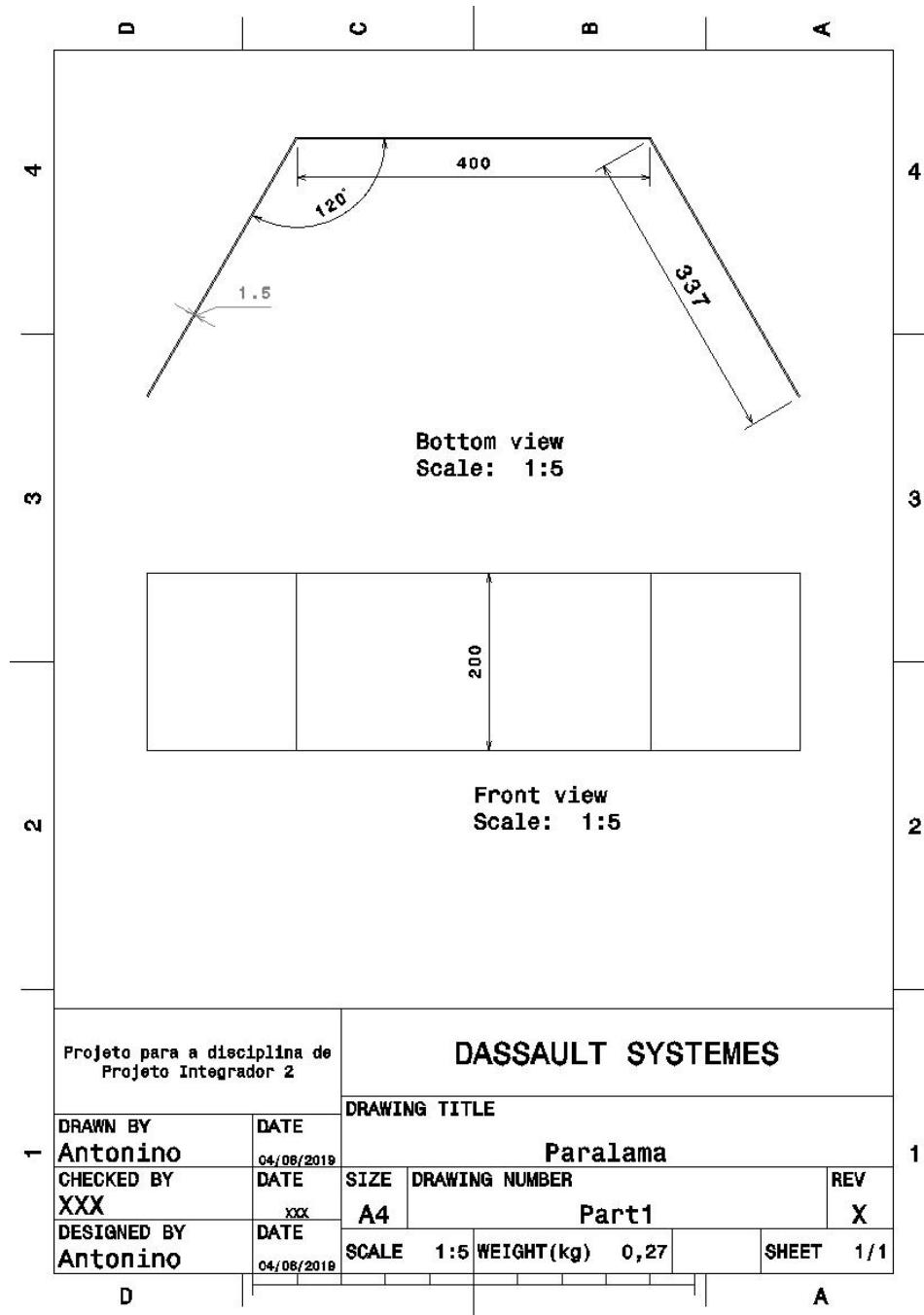


Figura 32 – Paralama

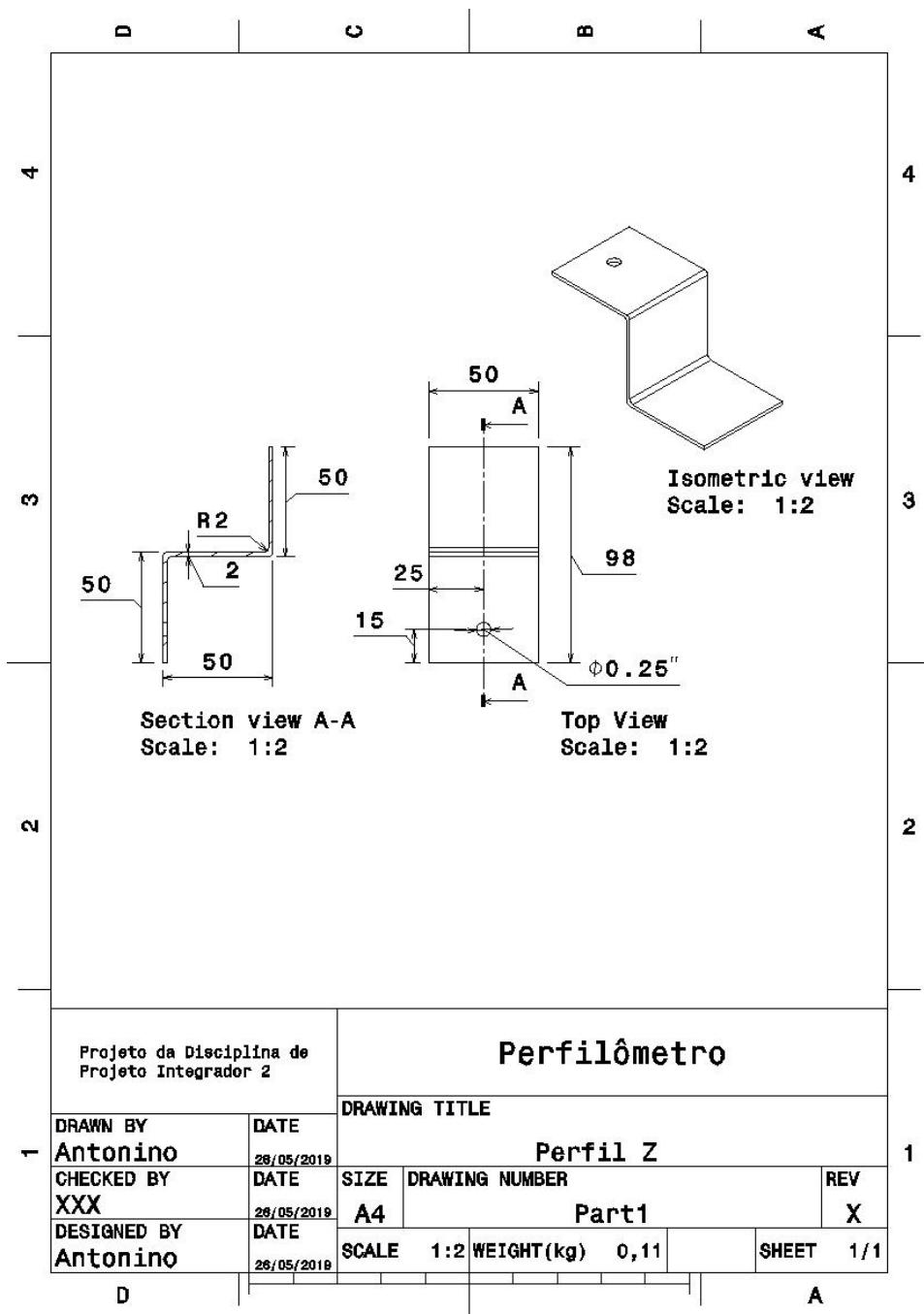


Figura 33 – Perfil em Z

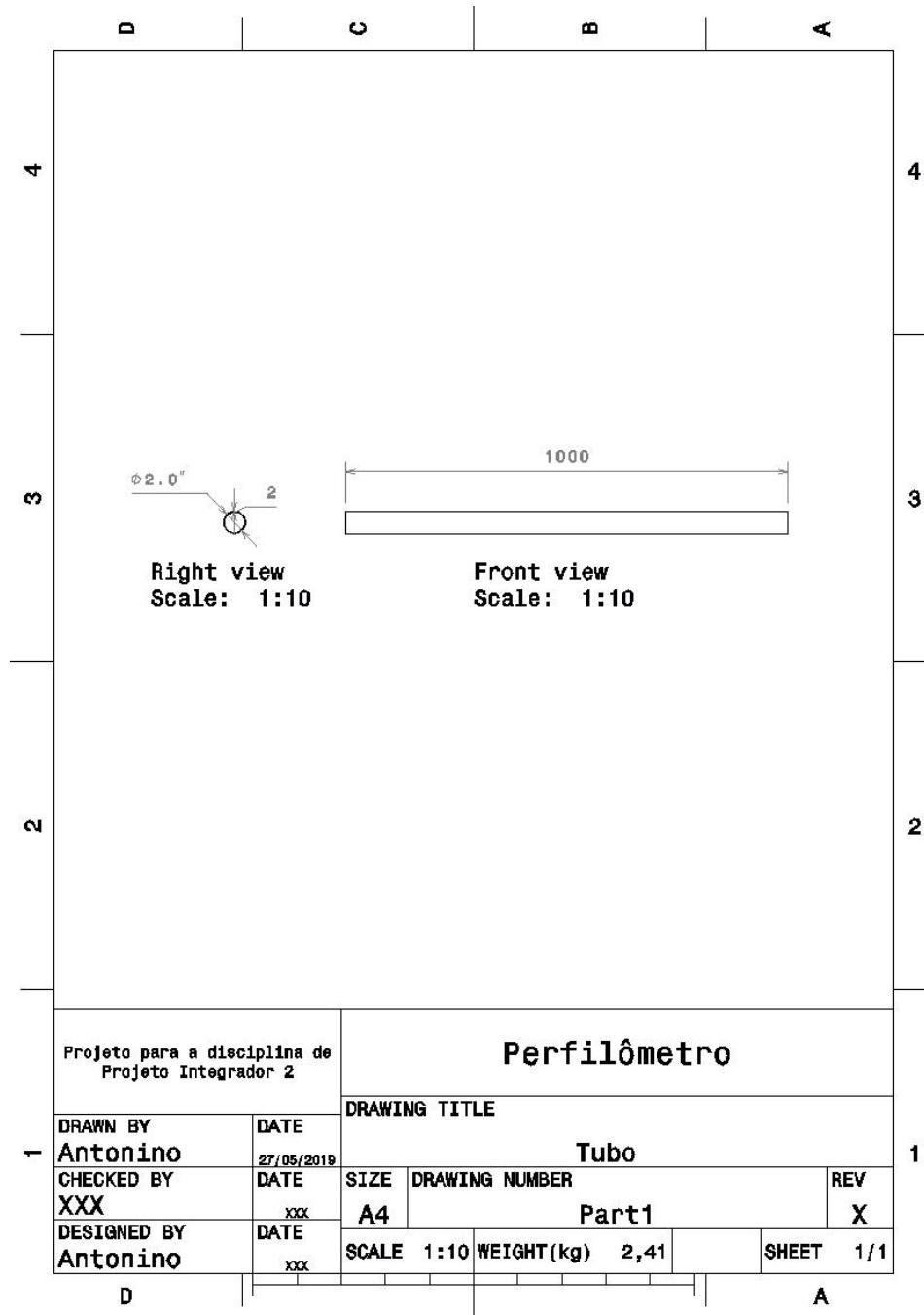


Figura 34 – Tubo circular no cambão

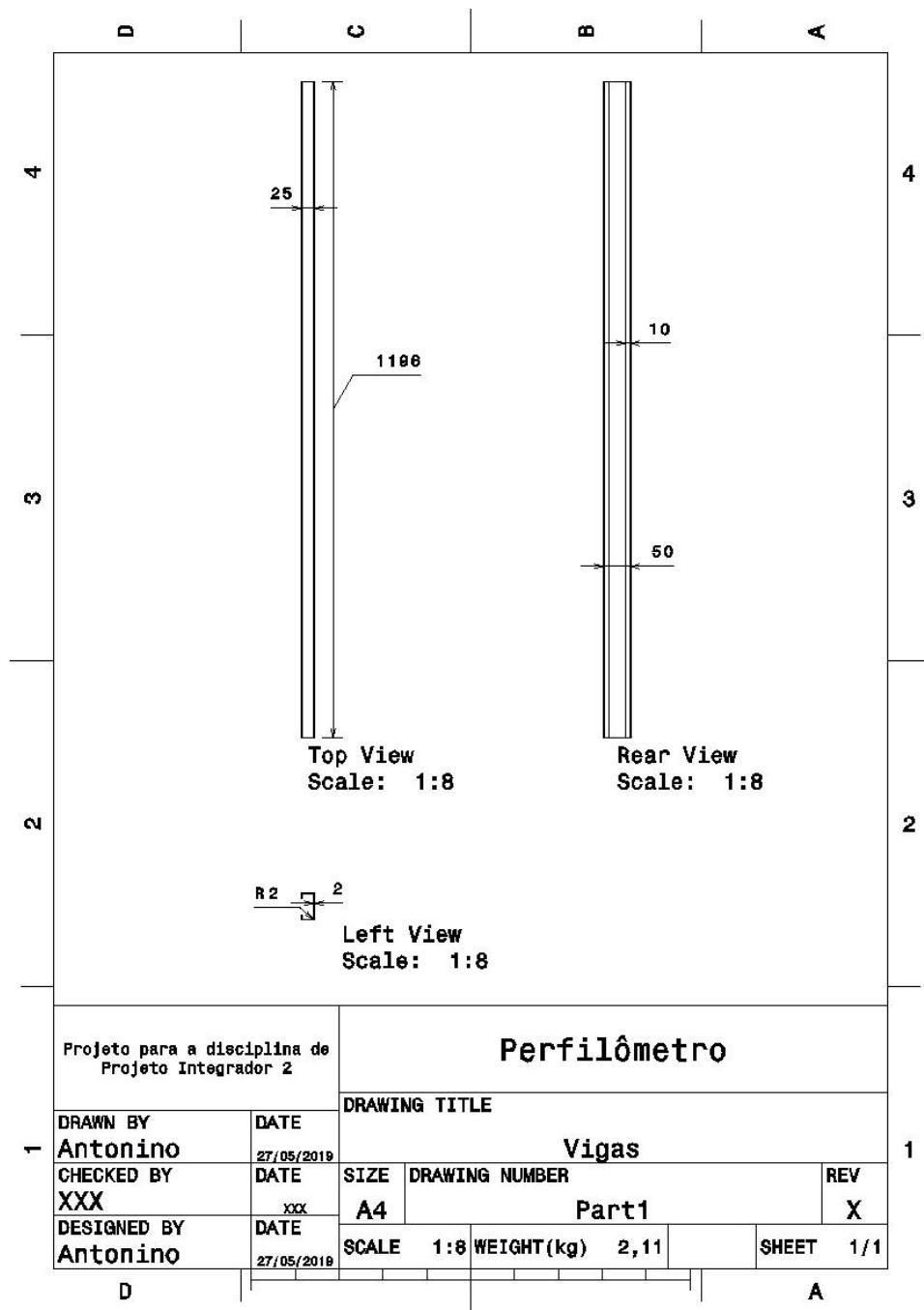


Figura 35 – Vigas

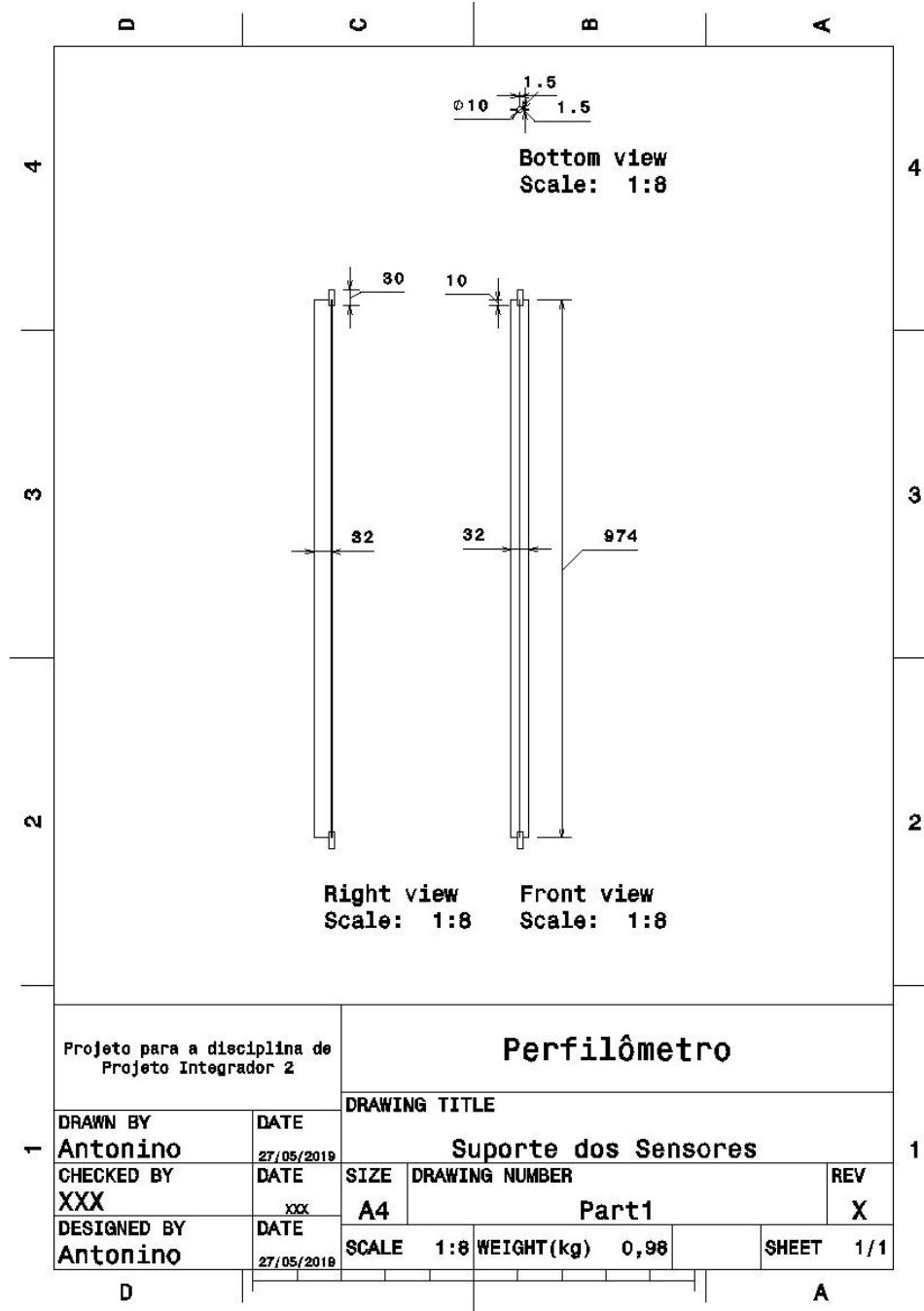


Figura 36 – Vigas dos sensores

3.3.1.2 Material Utilizado

A escolha de estruturas metálicas em aço são mais leves e resistentes, além disso, aliviam as cargas nas fundações e garantem uma instalação mais rápida e precisa.

O material utilizado no chassi do reboque sob análise foi o aço SAE 1020 com as propriedades encontradas no site do fabricante listadas abaixo:

Aço 1020	
Densidade	7,83g/cm ³
Resistência a tração	420 MPa
Resistência ao escoamento	350 MPa
Módulo de elasticidade	205 GPA
Coeficiente de Poisson	0,29

O aço SAE 1020 geralmente aplicado em peças de mecânica, são caracterizadas pela combinação de força e alta ductilidade, que é a habilidade do material em ser dobrado ou moldado. Podem ser soldadas facilmente através de métodos normais de soldagem.

3.3.1.3 Modelo Analítico

A metodologia analítica de cálculo dos esforços presentes na estrutura foi avaliada em relação ao equacionamento dos esforços externos gerados em cada parte da estrutura utilizando as equações básicas da estática dos sólidos esboçadas abaixo:

Em função das reações de apoio encontradas foi estimado os gráficos dos esforços internos ao longo de cada parte da estrutura do reboque. Para estruturas simples em que o número de equações em relação aos graus de liberdade são iguais ao número de incógnitas (estrutura isostática), a resolução pode ser calculada facilmente pelo o método analítico. Já para estruturas do tipo hiperestática isso não ocorre, pois é necessária a utilização de métodos numéricos para maior simplicidade dos resultados. A partir da estimativa das reações de apoio aplicadas na estrutura é possível encontrar as tensões cisalhantes e normais em função dos esforços internos que atuam na estrutura do reboque.

3.3.1.4 Modelo de Elementos Finitos

O método de elementos finitos (MEF) é uma técnica numérica para resolver problemas de valor limite nos quais um grande domínio em um meio contínuo é dividido em partes ou elementos menores. A solução é dada por meio equações diferenciais parciais em cada elemento levando em consideração a teoria da resistência dos materiais para o cálculo de deformações, tensões e deslocamentos como se cada elemento fosse um sistema isolado. Deve - se ressaltar que a solução de cada elemento depende do elemento vizinho e das condições de contorno imposto a ele, gerando assim um conjunto de equações algébricas simultâneas. Os método se divide basicamente em:

Discretização da estrutura Seleção de função de deslocamento adequada Encontrar as propriedades do elemento Montar as propriedades do elemento Aplicar as condições de contorno Resolver o sistema de equações Computação de resultados adicionais

Para simulação a estrutura em um software de elementos finitos foi utilizado o Ansys com o CAD importado do Catia.

3.3.1.5 Análise estrutural - Modelo de Análise

Como dito anteriormente, foi utilizado o aço 1020 para toda estrutura. Foram utilizadas dois tipos de vigas:

Perfil em U Enrijecido

Utilizado no chassi, o perfil em U enrijecido apresenta um alto momento resistor e menor concentrador de tensão. Foram usadas duas dimensões no reboque

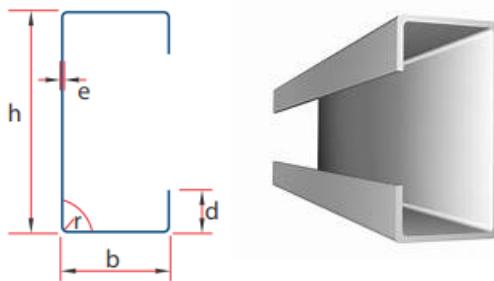


Figura 37 – Perfil U enrijecido

Propriedades da seção transversal - Perfil 1							
h	b	e	d	Wx	Jx	Wy	Jy
50 mm	25 mm	2mm	10 mm	3,08 cm ³	7,69 cm ⁴	1,10 cm ³	1,74 cm ⁴
Propriedades da seção transversal - Perfil 2							
h	b	e	d	Wx	Jx	Wy	Jy
75mm	40mm	2mm	10mm	7,93 cm ³	29,72 cm ⁴	3,09 cm ³	7,74 cm ⁴

Perfil tubular

Esse perfil foi utilizado no cambão

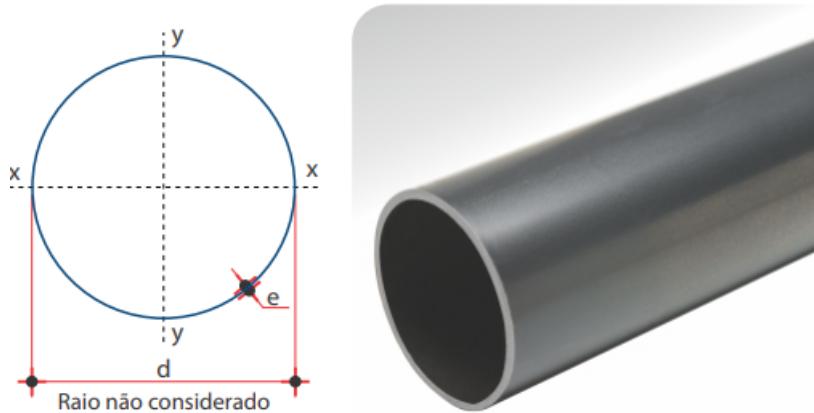


Figura 38 – Perfil circular

O cambão de perfil circular em reboques é normalmente utilizado por resistir melhor as tensões torcionais.

Para as vigas, cantoneiras e placas planas utilizadas nas maioria das imagens mostradas no desenho técnico foi feita pelo material de uma chapa plana de 1.5mm x 1200mm x 3000m. Os cortes que foram utilizadas na chapa estão abaixo:

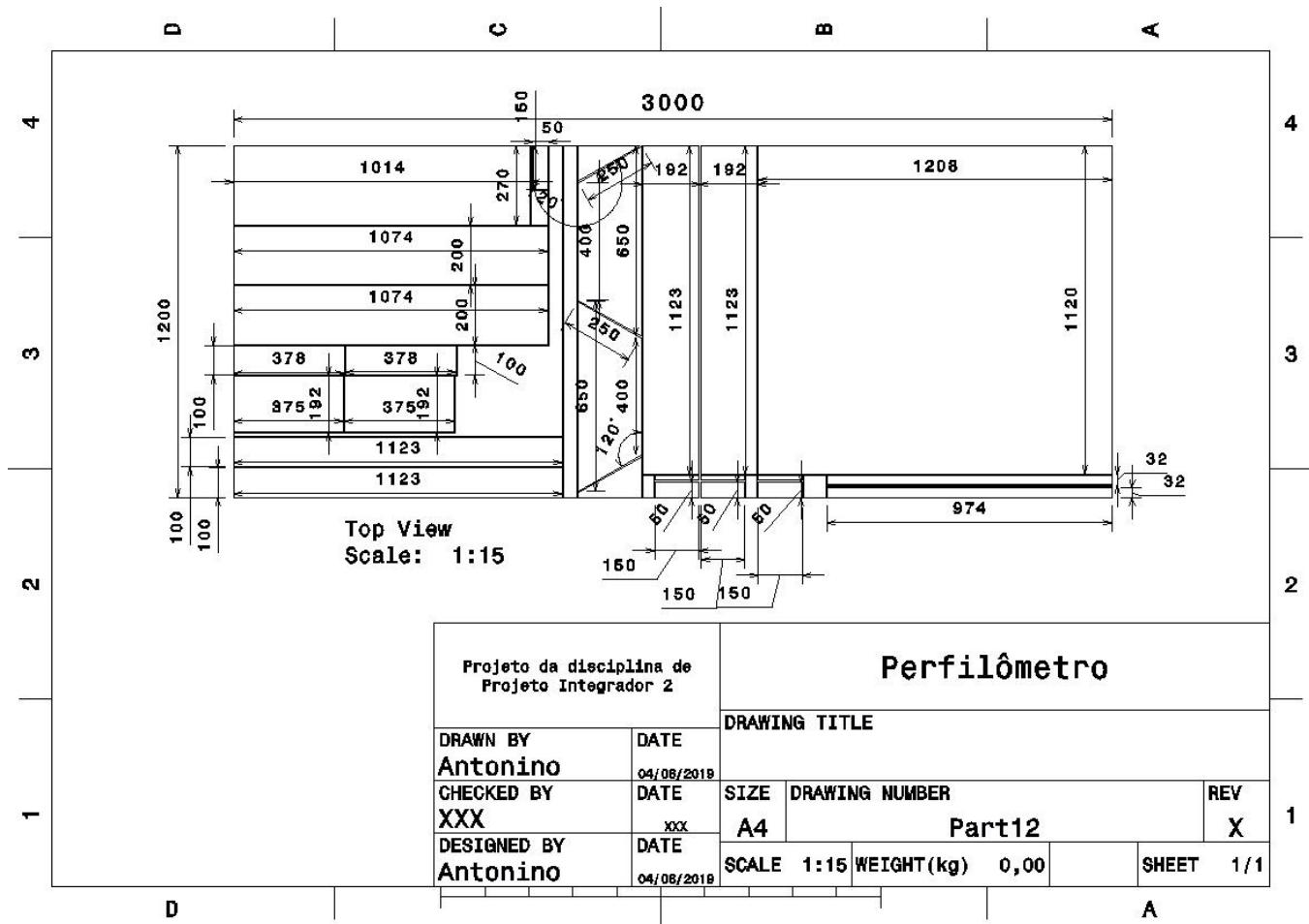


Figura 39 – Cortes realizados na chapa

3.3.1.6 Simulação

Para que as propriedades do reboque em relação a carga distribuída pudessem ser estimadas foi produzida uma malha no ambiente do software Ansys com a quantidade de elementos especificados na tabela abaixo:

Propriedades da Malha	
Quantidade de nós	44141
Quantidade de elementos	45221

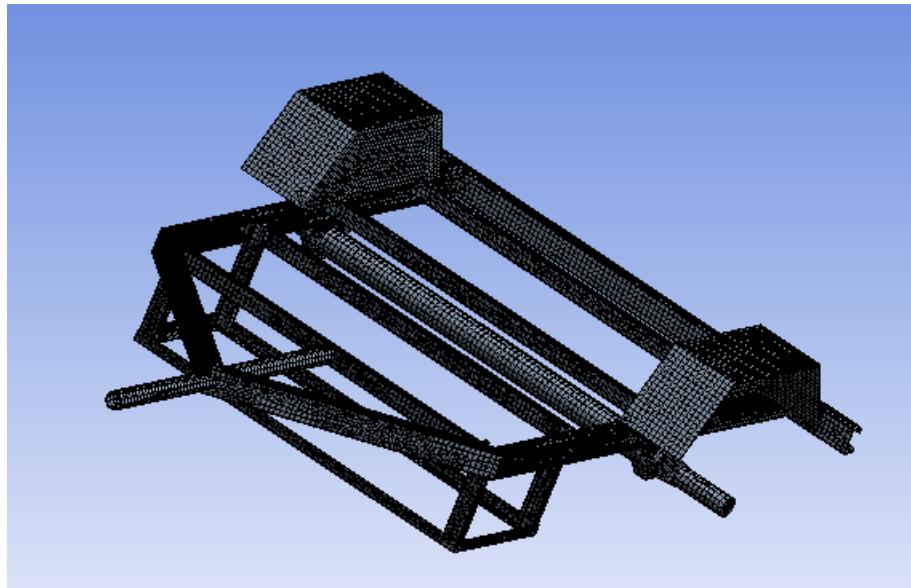


Figura 40 – Malha estruturada

O número de elementos na malha é um parâmetro que computa os formatos dos elementos para que as tensões e deformações presentes na estrutura sejam especificados por meio da aproximação utilizando conhecimentos na área de métodos numéricos utilizados pelo o software.

Após a criação da malha se faz necessário as reações de apoio no chassi. Foram considerados nas simulações um modelo de 6 graus de liberdades. Colocou- se uma restrição de movimento nos eixos X,Y e Z na extremidade da cambão mas com possibilidade de rotação nos três eixos. Já no engaste do chassi com o eixo de suspensão, foram restringidos os movimentos na posição X e Z e nas rotações dos três eixos.

Utilizou-se uma carga distribuída de 1 N chassi. Obteu-se em todo compartimento de carga do chassi como mostrado na imagem abaixo:

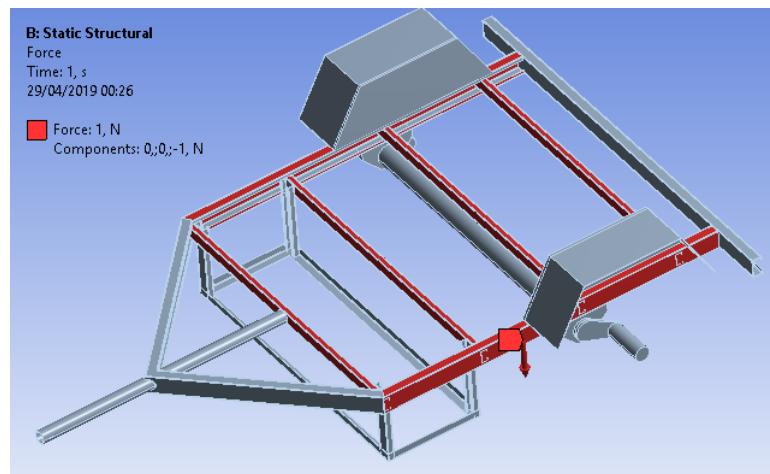


Figura 41 – Força distribuída aplicada

Obtêm-se os resultados de simulação para condições de contorno citadas acima:

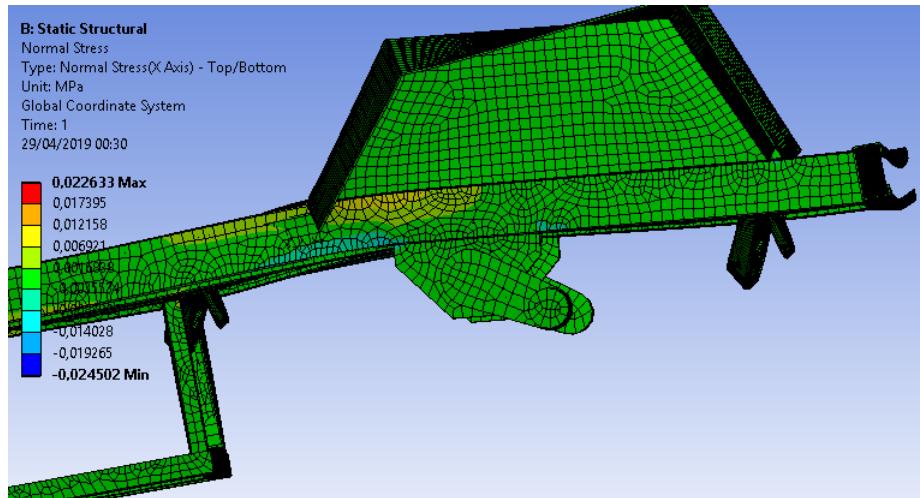


Figura 42 – Carregamento distribuída aplicado

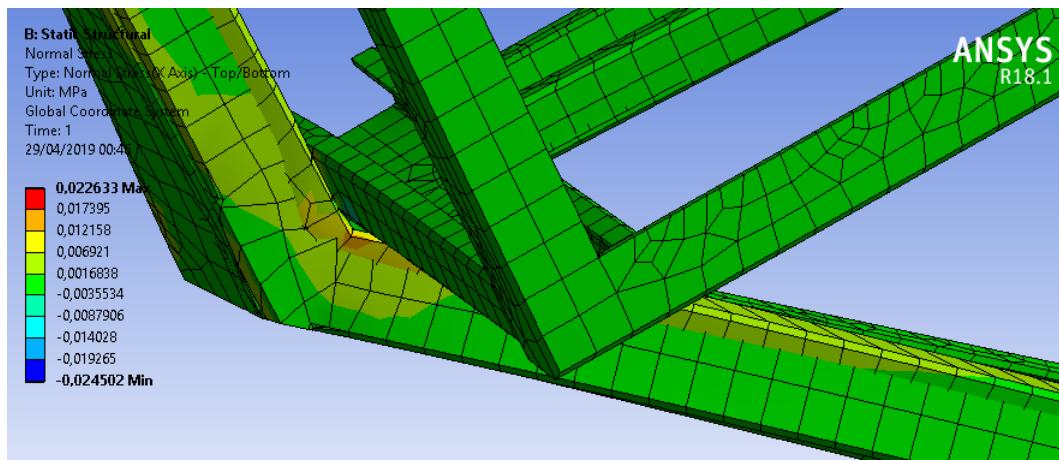


Figura 43 – Concentrador de tensão do cambão e o quadro

Constatou-se que os locais mais solicitados em tensões no chassi estão nos apoios engastados do chassi com a suspensão e o onde há a ligação do cambão com o quadro coincidindo com a região onde há mais ocorrência de falhas estruturais. Observa-se também que a estrutura apresenta como um todo um estado de tensão semelhante pois algumas partes mais solicitadas que outras em grande escala pode comprometer a estrutura.

Em função das análises das simulações especificadas e referente as tensões normais estimadas em relação ao seu valor máximo e mínimo de acordo com análise computacional. O software estimou se a estrutura permanece na zona elástica utilizando o critério de falha de Von Mises em que em relação ao conhecimentos de estática dos sólidos tem uma margem maior para o resultado da análise de tensões que não pode exceder a tensão de escoamento a formulação e esboçada abaixo.

$$\sqrt{\sigma_1^2 - \sigma_1\sigma_2 + \sigma_2^2} < \sigma_Y \quad (3.15)$$

Na formulação a tensões utilizadas são tensões geradas por forças axiais que no estado plano de tensões representam as tensões principais do círculo de mohr, como o valor encontrado no critério de von mises é menor do que a tensão de escoamento do material o mesmo encontra - se na zona elástica portanto, o fator de segurança está dentro dos requisitos do projeto. A figura abaixo esboça o processo .

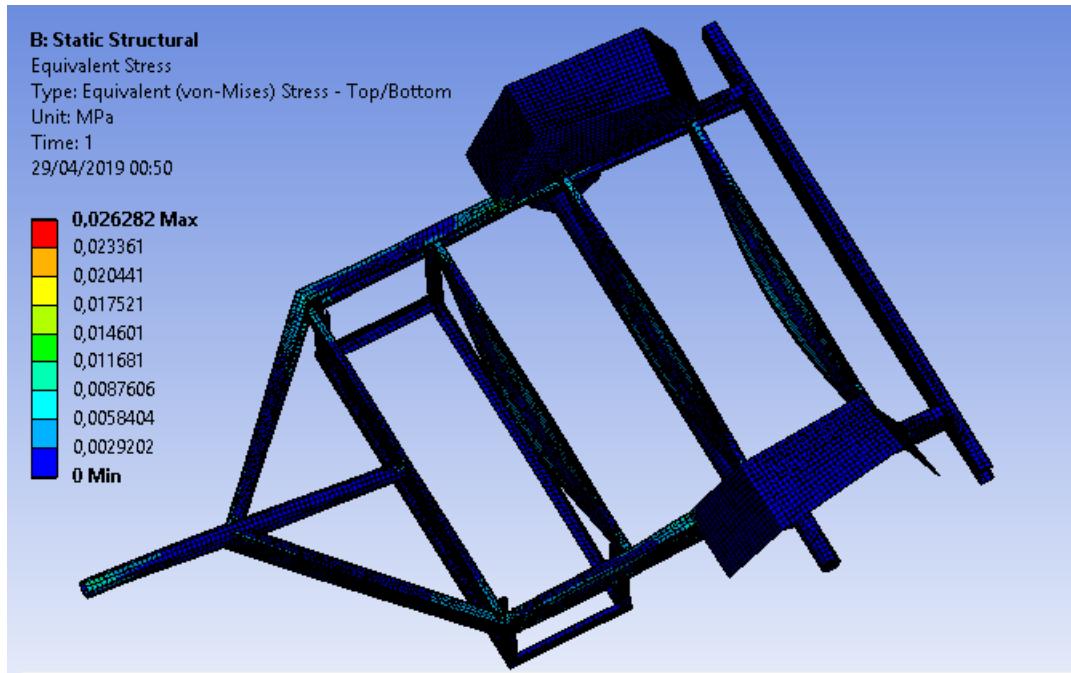


Figura 44 – Tensões equivalentes utilizando o critério de Von Mises

3.4 Software

O código e a documentação do software estão no GitHub em três repositórios da organização PI2:

- Frontend Swift: <https://github.com/pi2-fga/201901-Perfilometro-Frontend>
- BackEnd NodeJS: <https://github.com/pi2-fga/2019-Perfilometro-Backend>
- BackEnd Python: <https://github.com/pi2-fga/201901-Perfilometro-Python-Backend>

3.4.1 Tecnologias Utilizadas

- XCODE 10: O Xcode é o ambiente para desenvolvimento de aplicações Apple. Atualmente, é uma das IDE's mais completas, apresenta ferramentas de source control(GIT), teste unitário, monitoramento de perfomance, testes unitários e deploy de aplicações iOS([APPLE, 2019d](#));
- Swift 5.0: O Swift é uma linguagem Open Source criada em 2014 pela Apple com intuito de facilitar o desenvolvimento de apliacaões iOS. A linguagem teve um crescimento enorme nos últimos anos baseado em quatro princípios: Paralelismo, Facilidade, velocidade e Flexibilidade. Atualmente, encontra-se na versão 5.0, sendo que anualmente é disponibilizada uma nova versão([APPLE, 2019c](#));
- Core Bluetooth: É um framework disponibilizado pela Apple responsável por realizar a comunicação com dispositivos bluetooth de baixa energia. Devido a estas características ele é amplamente utilizado em projeto de IoT(Internet of Things)([APPLE, 2019a](#));
- Core Location: É um framework da Apple que permite determinar informações geográficas tais como orientação, posição, orientação e posição relativa a objetos(iBeacon). Ele utiliza de todos os recursos disponíveis no dispositivo iOS para ter acesso a estes dados([APPLE, 2019b](#));
- Google Maps API: API da Google que permite acesso a mapas do Google Maps e seus dados. É a API com a maior base de dados e recursos disponível atualmente([GOOGLE, 2019](#));
- Firebase: Firebase é uma plataforma da Google que atualmente apresenta mais de 15 produtos diferentes. Neste projeto, pode-se citar alguns dos serviços do Firebase a serem utilizados, tais como: RealTime database; e Firebase Storage; Atualmente, é amplamente utilizado devido a facilidade de configuração e diversidade de funcionalidades para o desenvolvimento mobile([FIREBASE, 2019](#));

- Charts: Atualmente, Charts é o Framework de gráfico mais utilizado no desenvolvimento iOS. Ele apresenta mais de 8 tipos diferentes de gráficos e uma documentação completa. Atualmente, ele possui mais de 20 mil estrelas no Github([GINDI, 2019](#)).

3.4.2 Diagramas

3.4.2.1 Diagrama de Classes

Devido as características do projeto, o time optou por escolher o processo de desenvolvimento ágil com objetivo de atender as demandas e qualidades. Dessa forma, a equipe criou e continua a desenvolver novas versões para modelos de arquitetura. Por ser um projeto que está em constante evolução o time criou modelos arquiteturais mais gerais, visto que a todo momento módulo, classes e componentes estão se alterando.

3.4.2.1.1 Diagrama de Classe - MVP - Model View Presenter (0.1)

Modelo que separa a arquitetura do frontend mobile em Model onde estão os modelos de dados, View onde estão presentes os componentes de UI e interação com o Usuário e Presenter camada por gerenciar regras de negócio e formatação de dados. Este modelo é considerado mais testável em relação ao MVC, visto que apresenta uma camada apenas para formatação da controller.

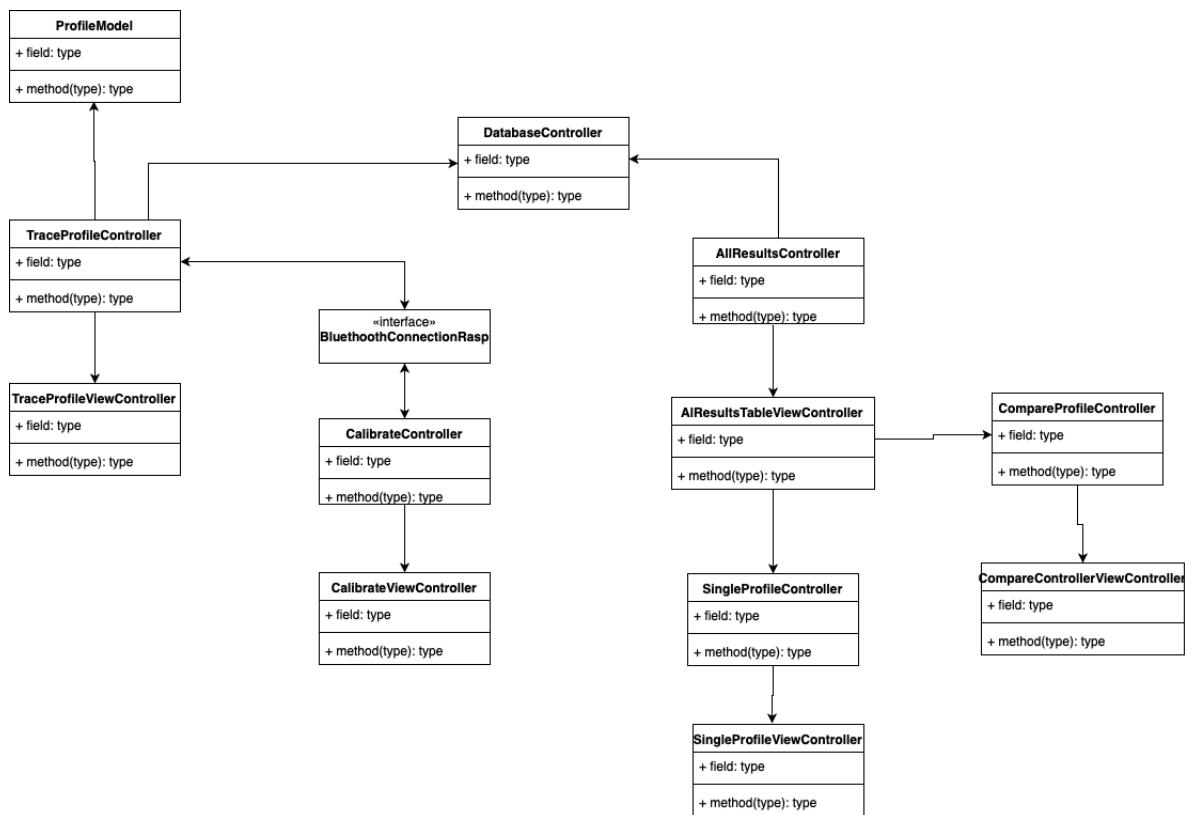


Figura 45 – Diagrama de Classe - Versão 0.1

3.4.2.1.2 Diagrama de Classe - VIP - View Interactor Presenter (1.0)

O VIP foi proposto a partir dos princípios da *Clean Architecture* postulada por Uncle Bob. Ela é compostas por cenas que consiste numa unidade reduzida de funcionalidade, por exemplo autenticação, configuração, visualizar perfil, entre outras. As cenas possuem módulos de formatação dos dados e interação com o usuário. Também, tem uma camada de modelos de dados, conhecida como model, ela também é responsável por formatar os dados entre as camadas. Por fim, existem protocolos conhecidos como *boundaries* que fazer a comunicação entre os módulos, a seguir segue uma breve representação.

Ressalta-se ainda que cada cena do VIP apresenta as seguintes camadas: View, Model, Interactor, Worker e Presente. Este fato auxilia na divisão de responsabilidades, manutenção e testabilidade. Dessa forma, as cenas representam conjuntos de classe. Para facilitar a visualização as cenas estão representadas em pacotes, visto que cada cenas apresenta os módulos já descritos.

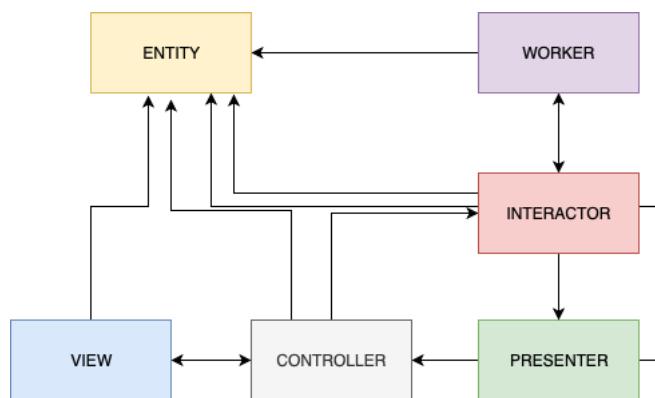


Figura 46 – Estrutura de Cena - VIP

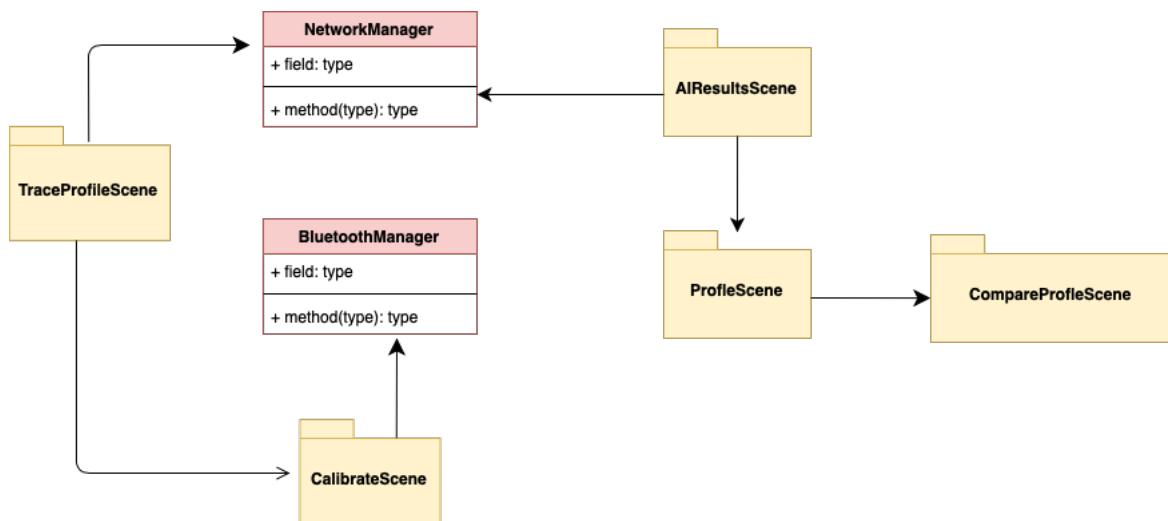


Figura 47 – Diagrama de Classe - Versão 1.0

3.4.2.2 Microserviços e Frameworks

Com objetivo de realizar a manipulação com outros componentes, foi criada um servidor central em NODE contendo vários serviços. Dentre eles, alguns serviços do firebase, tal como o banco de dados. Ressalta-se que a maior parte do processamento será feita em nível de frontend. Nos, quais sera necessária a comunicação entre componentes de hardware e celular via API e/ou Bluetooth.

3.4.2.2.1 Versão 1.0

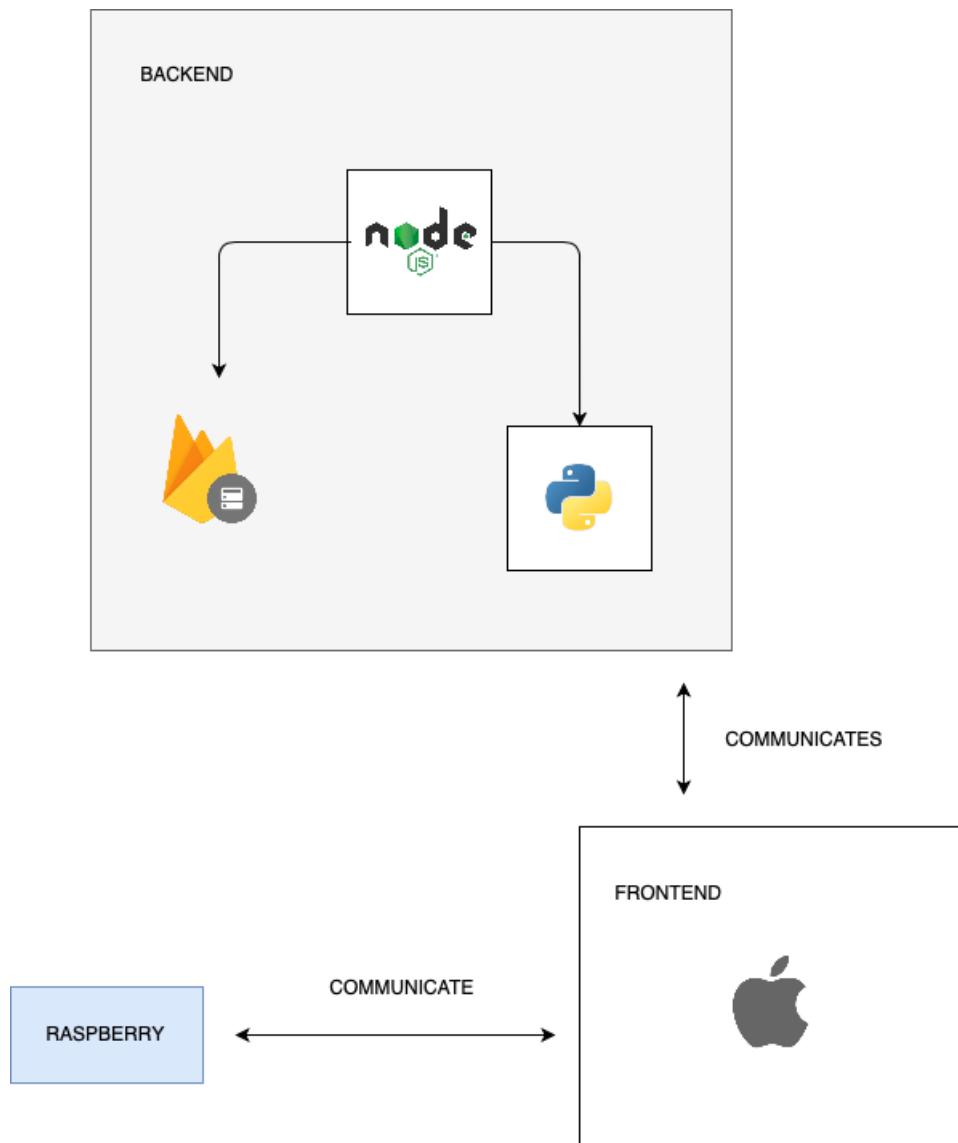


Figura 48 – Micro Serviços - Versão 1.0

3.4.2.2.2 Versão 2.0

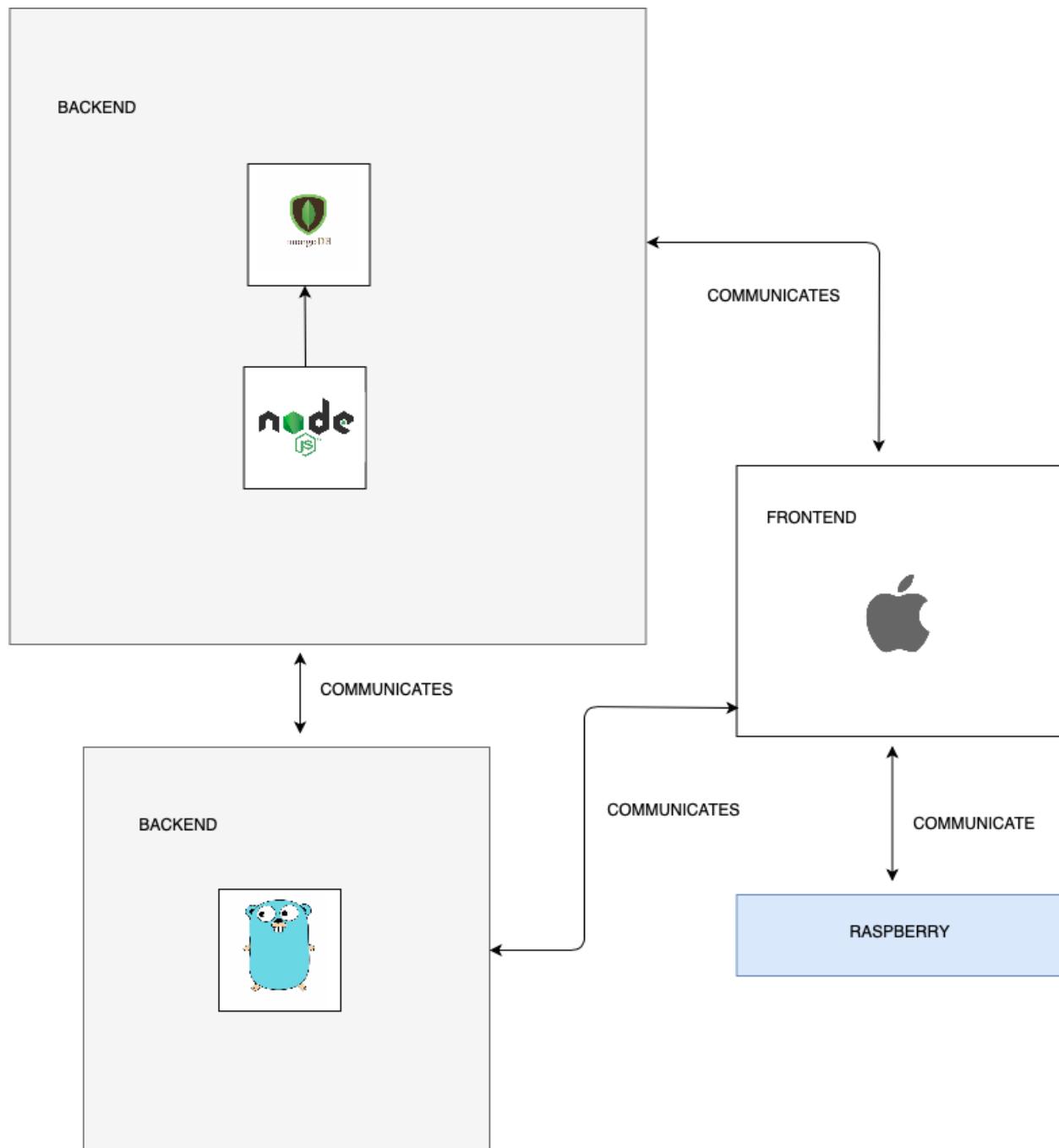


Figura 49 – Micro Serviços - Versão 2.0

3.4.2.3 Diagrama de Caso de uso

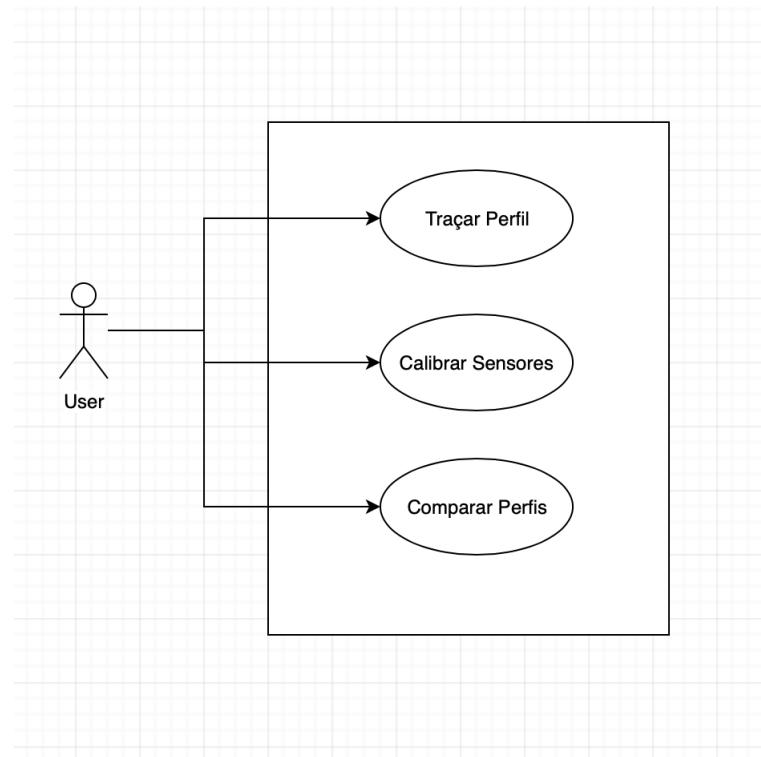


Figura 50 – Diagrama de caso de uso

3.4.2.4 Diagrama de componentes

O diagrama de componentes do sistema pode ser visto na figura 51

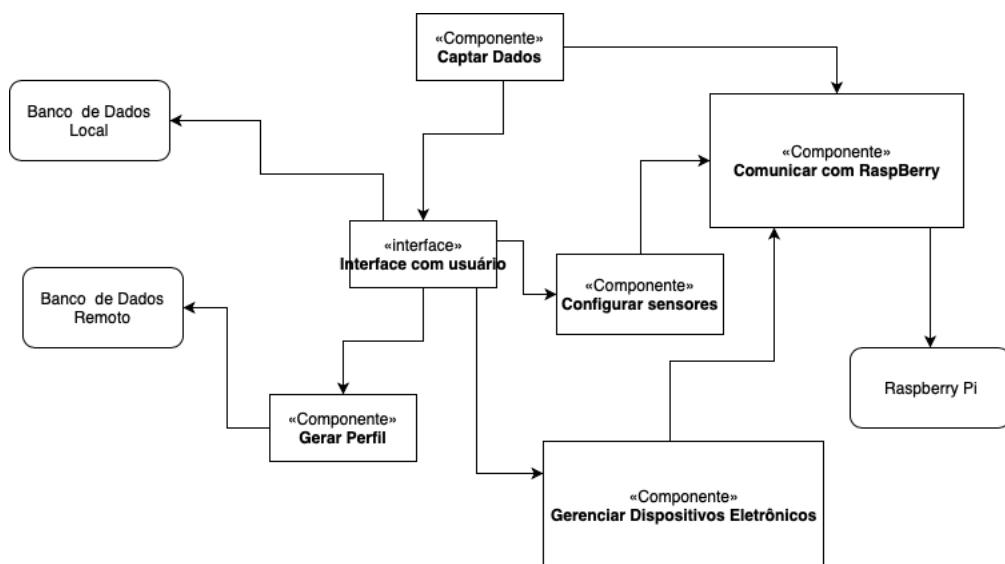


Figura 51 – Diagrama de componentes

3.4.3 Arquitetura Backend

O servidor principal tem como finalidade realizar a comunicação com o Firebase, banco de dados da aplicação, e fazer o controle das ações provenientes do *frontend*. Em níveis arquiteturais espera-se fragmentar o servidor por camadas, router, model e controller, de modo que elas dividam suas responsabilidades. Apesar de não existir o conceito de classe, espera-se fragmentar a aplicação em três camadas ou mais com objetivo de tornar mais fácil a manutenção e evolução desse serviço.

Não obstante, neste processo, utilizou-se dos seguintes frameworks até o momento:

- Nodemon: Framework que facilita o processo de desenvolvimento, visto que a partir do salvamento de qualquer alteração de código, ele reinicia o servidor, automatizando este processo.
- Restify: Uma das maiores e mais estáveis APIS para rotas em node
- Lodash: Framework que dispõe de inúmeras ferramentas no desenvolvimento com Ecmascript.

3.4.4 Arquitetura Framework Gráfico em três Dimensões

Não foi encontrado um framework de gráfico em três dimensões para iOS. Dessa forma a equipe se propôs a criar um framework para essa finalidade. Este será orientado a protocolos, assim com o Swift 5.0 e também fará uso do SceneKit.

3.4.5 Política de Branches - GitFlow

Com objetivo de organizar o fluxo de trabalho, definiu-se uma política de branches, da seguinte forma:

Branch master: Branch que contém código em nível de produção, no caso deste projeto é a branch que conterá o código para ser apresentado para as releases. O código mais maduro existente na sua aplicação. Todo o código novo produzido eventualmente é juntado com a branch master, em algum momento do desenvolvimento.

Branch hotfix: São branches no qual são realizadas correções de bugs críticos encontrados em ambiente de produção, e que por isso são criadas a partir da branch master, e são juntadas diretamente com a branch master e com a branch dev. Essas branches deverão ter o nome começando com a palavra "hotfix/" e terminado com a ultima tag da branch master. Ex: hotfix/0.1

Branch dev: Branch que contém código em nível preparatório para a próxima release. Quando features são terminadas, elas são juntadas com a branch dev, testadas e

somente depois as atualizações da branch dev passam por mais um processo para então ser juntadas com a branch master.

Branch feature: Branches quais são desenvolvidos recursos novos para o projeto. Elas tem por são criadas a partir da branch dev (pois um recurso pode depender diretamente de outro recurso em algumas situações), e, ao final, são juntadas com a branch dev via pull requests. Essas branches deverão ter o nome começando com a palavra "feature/" e terminado com o número referente a issue a qual a feature está associada. Ex: feature/14

3.4.6 Papéis de Atuação

Apesar do projeto já definir os responsabilidades e papéis, o time de software decidiu definir *roles* para cada um dos integrantes com o intuito de responsabilizar e dividir as responsabilidades entre essa subequipe. Deve-se destacar que apesar do papel definido, cada um dos membros pode realizar tarefas diferentes das definidas inicialmente.

3.4.6.1 Scrum Master

Ficou definidas as seguintes tarefas para o Scrum Master (Geovanni Oliveira):

- Fazer gestão de riscos;
- Garantir que os membros da equipe estejam executando a metodologia;
- Ficar responsável por conduzir as reuniões;
- Manter os *stakeholders* a par do projeto;
- Resolver ou designar membros da equipe para resolver impedimentos;
- Garantir que nos ritos os dados sobre a equipe e a *sprint* estejam disponíveis;
- Garantir que as reuniões sejam preparadas e que não demore mais que o tempo estipulado;

3.4.6.2 DevOps

Ficou definidas as seguintes tarefas para o DevOps (Guilherme Baldissera):

- Garantir que todas as alterações de código e configurações sejam feitas usando mecanismos automatizados, rastreáveis e repetíveis.
- Implantação contínua de mudanças do check-in para produção.
- Infraestrutura como código.

3.4.6.3 Arquiteto

Ficou definidas as seguintes tarefas para o Arquiteto (Miguel Pimentel):

- Limitar as escolhas dentro do desenvolvimento em relação: A padrões de projetos a serem utilizados; Definir/Criar frameworks a serem utilizados no processo de desenvolvimento.
- Identificar pontos de reutilização de código no desenvolvimento do projeto por meio: Enxergar de forma mais abrangente; projeto baseando-se em componentes; e ter contato com todos os produtos desenvolvidos ao longo do projeto.
- Definir escolhas em relação: A arquitetura ser fragmentada em pequenos formatos como forma de diminuir a complexidade no processo de desenvolvimento;
- Ter domínio em relação: Função componentes; dependência de componentes; e comunicar arquitetura ao time de desenvolvimento.

3.4.7 Protótipo de Papel

O protótipo de papel foi feito com o intuito de projetar o aplicativo de forma que nele as guide lines da Apple fossem respeitadas e visando a melhor compreensão do usuário ao utilizar o sistema de traçar o perfil da pista. Dessa maneira a facilidade de uso foi levada em consideração.

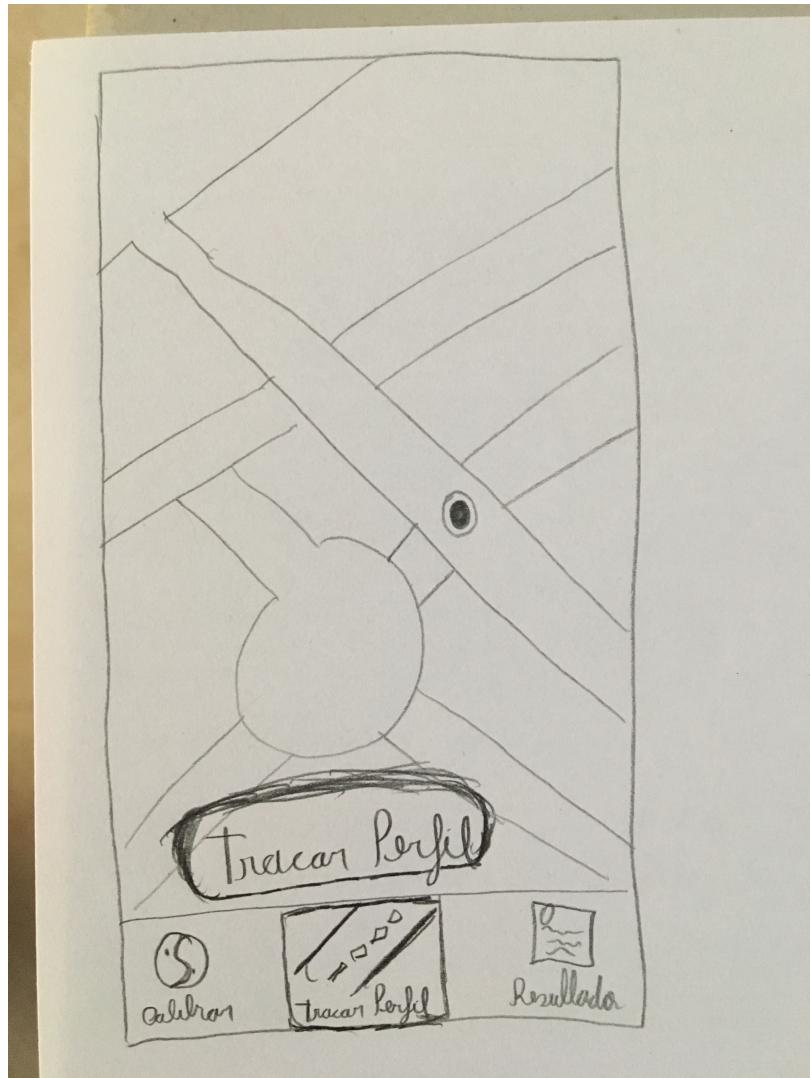


Figura 52 – Tela inicial do aplicativo do perfilômetro

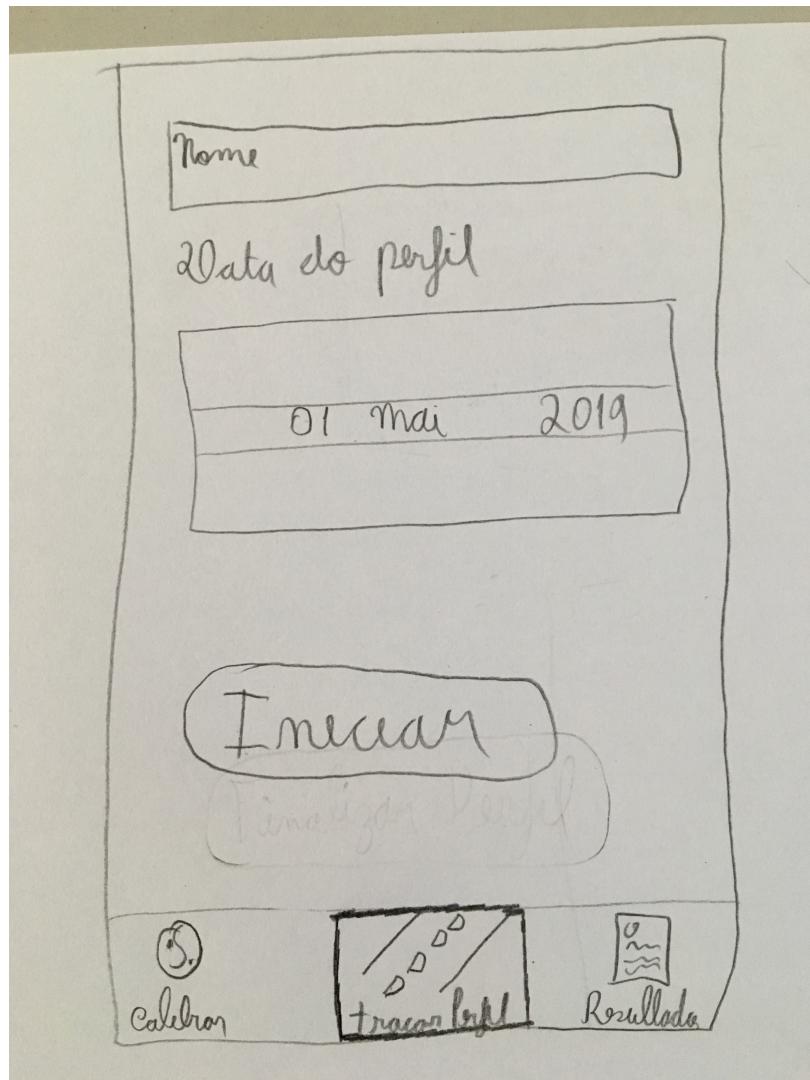


Figura 53 – Tela inicial de traçar novo perfil

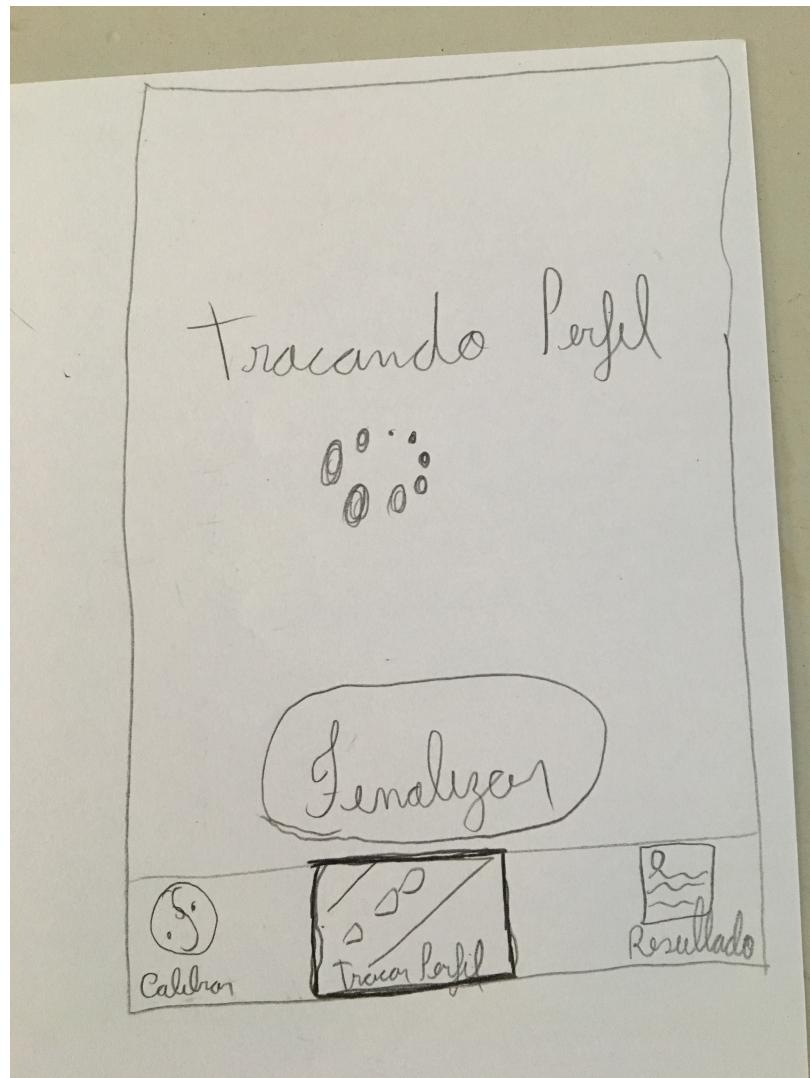


Figura 54 – Feedback traçando perfil

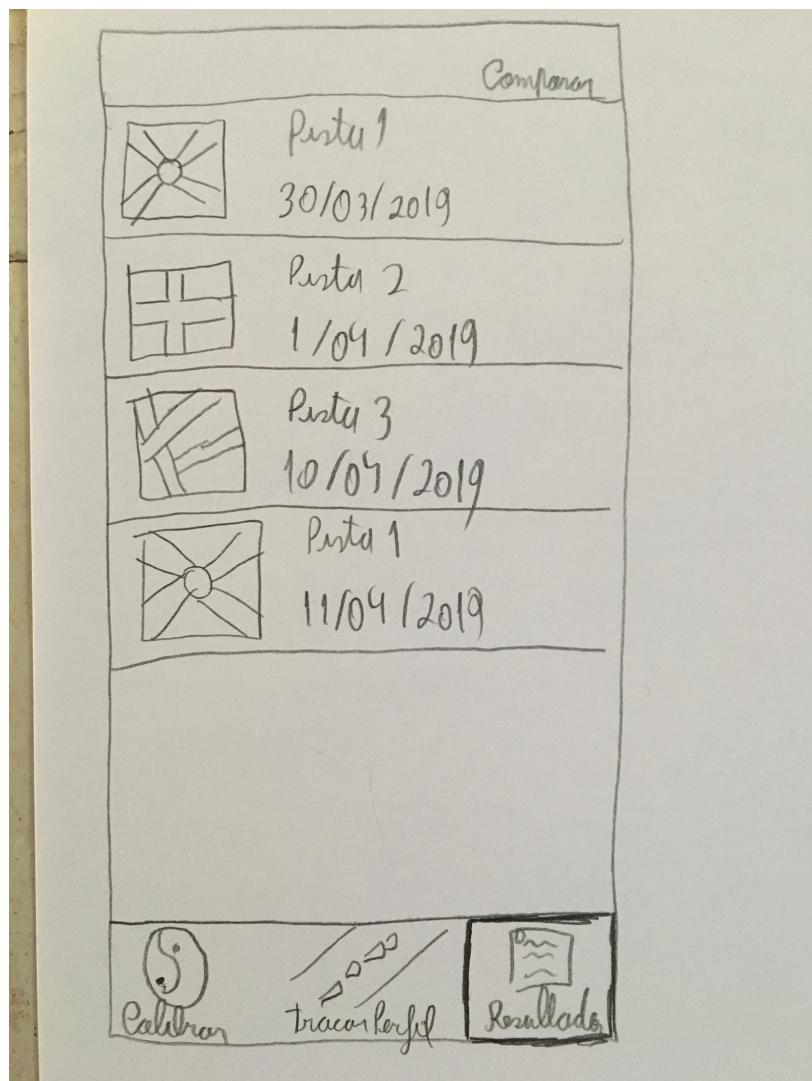


Figura 55 – Lista de perfis

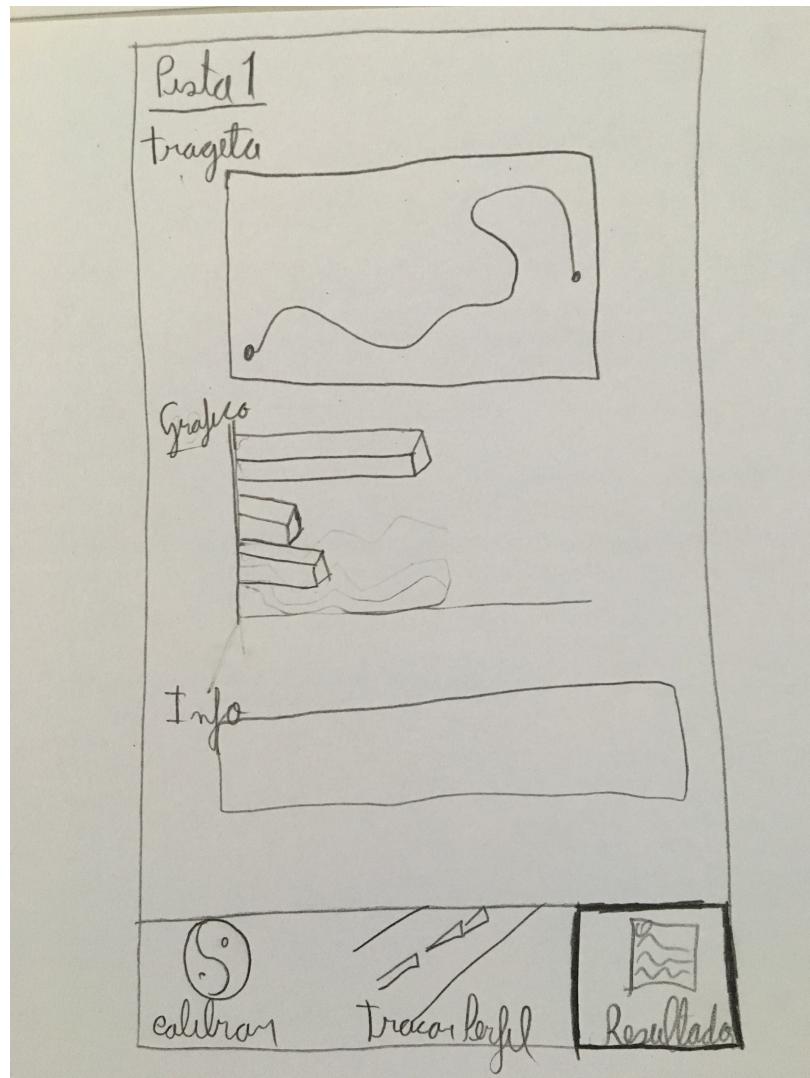


Figura 56 – Ver perfil

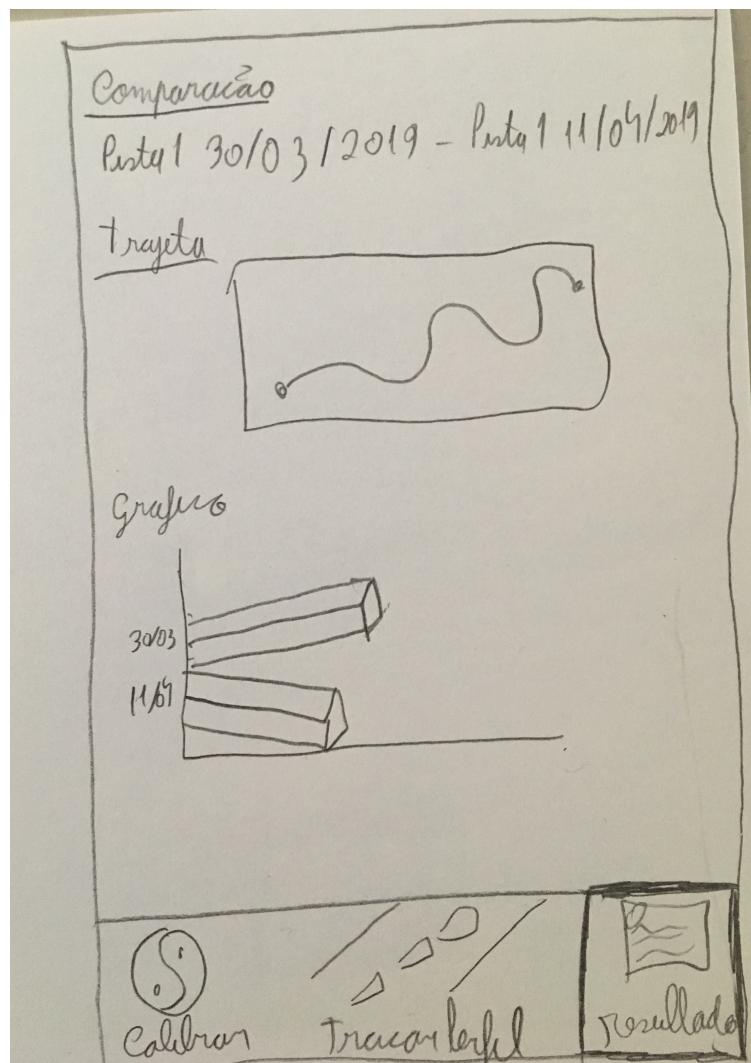


Figura 57 – Tela de comparação de perfis

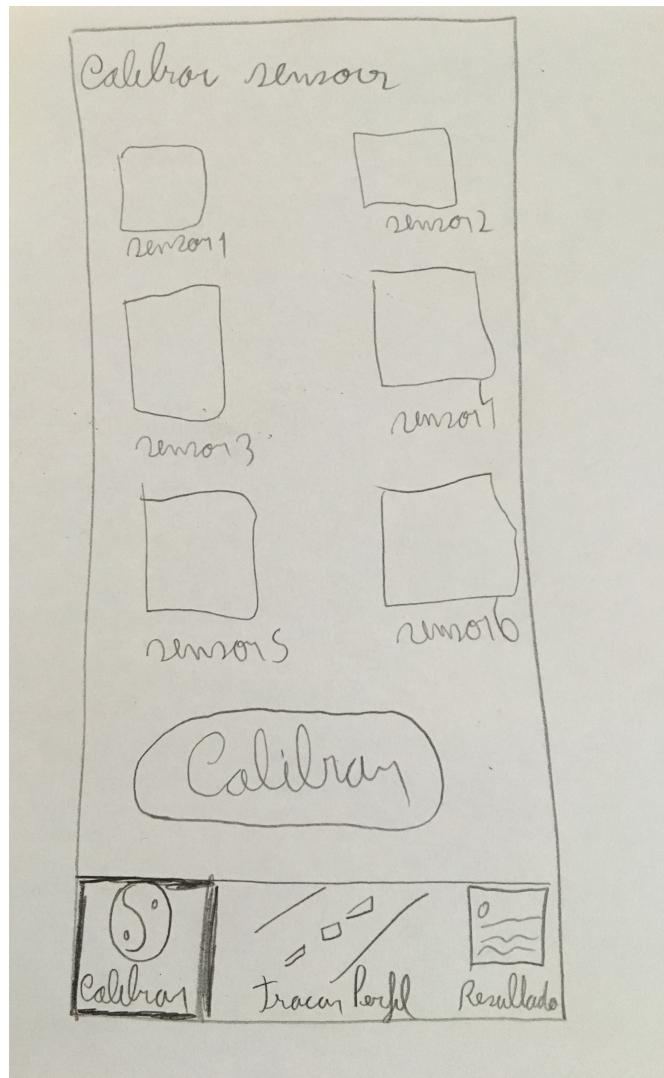


Figura 58 – Tela listando sensores

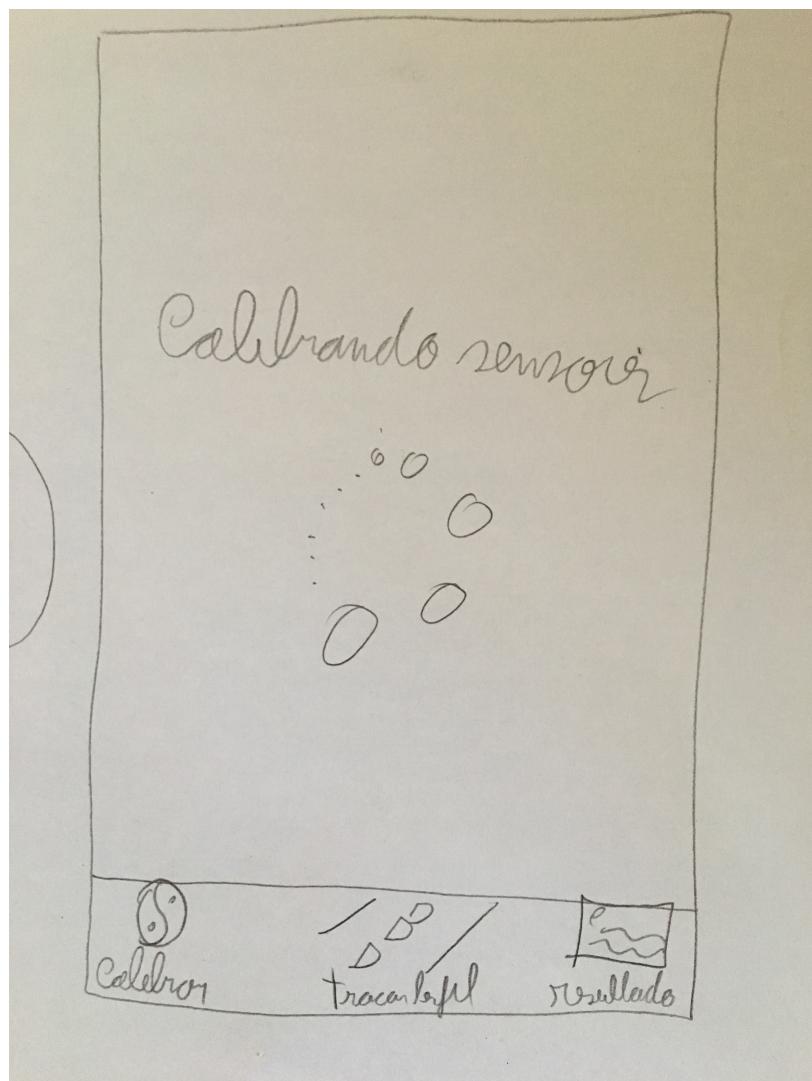


Figura 59 – Feedback de calibração dos sensores

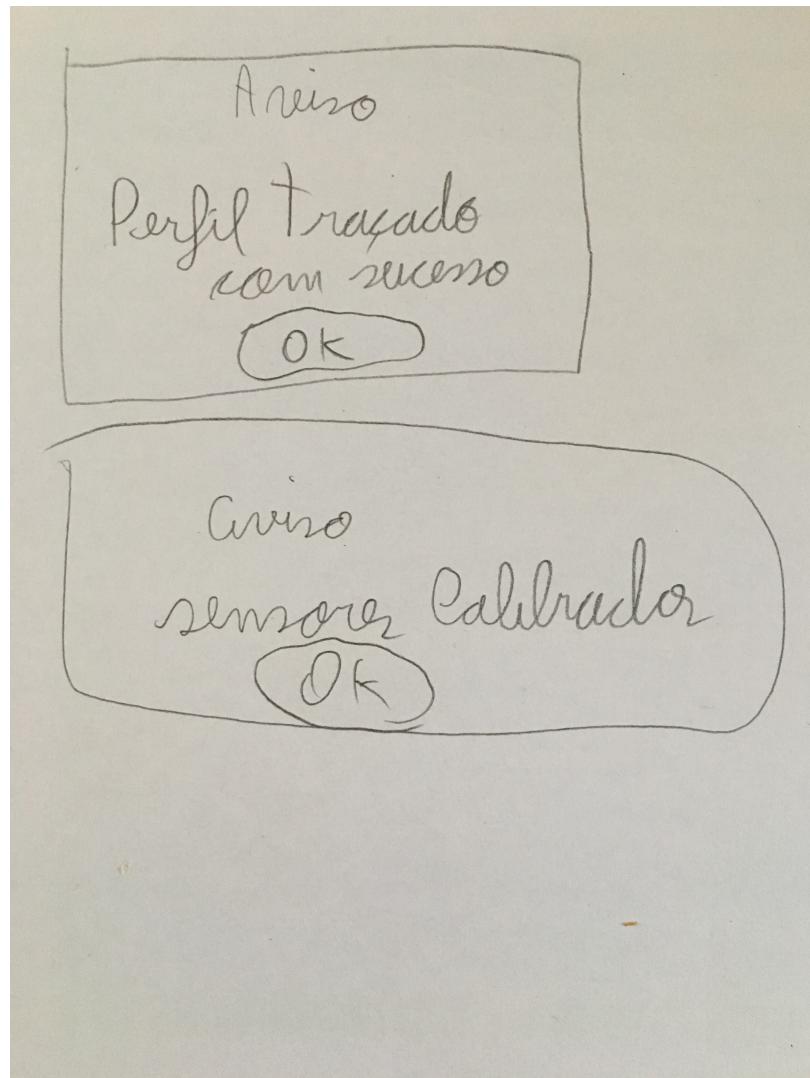


Figura 60 – Avisos ao usuário

3.4.8 Atividades Realizadas

Neste etapa entre os pontos de controle 2 e 3, ocorreram o desenvolvimento inicial de todos as nossas frentes, servidores (Go, NodeJS) e Aplicativo iOS (frontend). A tabela a seguir apresenta as modificações realizadas.

BACKLOG	TO DO	DOING	DONE
Teste de integração por bluetooth	Teste de software	Polimentos de UI - Frontend iOS	Desenvolvimento inicial
		Microserviço Go - Data Handler	Especificação de requisitos
			Estimativa de custos
			Diagrama de Classes
			Diagrama de fronteira
			Diagrama de Classes
			Diagrama de caso de uso
			Definir tecnologias
			Prototipação
			Diagrama de componentes

Tabela 4 – Backlog atual das atividades

Ressalta-se que o microserviço para cálculos matemáticos será implementado por último. Como pode se observar, a equipe terminou as funcionalidades referente ao aplicativo ou seja, comunicação HTTP, Bluetooth, Google Maps, Pesquisa de lugares. Entretanto, ainda existem aprimoramento em nível de UI que podem ser aplicados. Ressalta-se que a equipe mudou a escolha do uso do Firebase para o Mongo, visto que este se adapta mais a grandes volumes de dados numa mesma entidade(Orientado a documento).

4 Gerenciamento do Projeto

4.1 Cronograma e Sequenciamento de Atividades

O cronograma desenvolvido está sendo parcialmente seguido, porém sem grandes impactos no andamento do projeto. As estimativas de datas estão até o momento norteando as entregas e direcionando os integrantes em suas atividades. O cronograma completo está nesse [link](#).

4.2 Milestones Identificados

Dentro do cronograma, estão identificados milestones importantes que estão listados abaixo. Estamos atualmente no Milestone 3.

Título	Data de Entrega	Equipe
Ponto de Controle 1	03/04/2019	
Apresentação	12/04/2019	Geral
Ponto de Controle 2	03/05/2019	
Apresentação	15/05/2019	Geral
Ponto de Controle 3	05/06/2019	
Apresentação	14/06/2019	Geral
Ponto de Controle 4	10/07/2019	
Aplicação do perfilômetro em contexto real	26/06/2019	Geral
Apresentação	05/07/2019	Geral

Tabela 5 – Milestones

4.3 Custos atuais do projeto

Os custos até o momento para a execução do projeto estão listados na tabela abaixo e estão separados nas área de atuação das equipes: Estrutura, Software e Eletrônica e Energia.

	Produto	Valor Unitário	Quantidade	Total
	Estrutura			
	Tubo circular 2"x 2m x 6m	R\$ 78,44	1	R\$ 78,44
	Perfil em Aço "C"Enrijecido 50 x 25 x 10 x 2,00 mm 6m	R\$ 57,23	1	R\$ 57,23
	Perfil em Aço "C"Enrijecido 75 x 40 x 15 x 2,00 mm 6m	R\$ 70,46	1	R\$ 70,46
	Chapa de aço 1,5 mm x 1.2 m	\$ 228,51	1	R\$ 228,51
	Suspensão Eixo de Torçao 500 kg	R\$ 799,00	1	R\$ 799,00
	Pneu 175/70 R13	R\$ 150,00	2	R\$ 300,00
	Roda de ferro aro 13 cubo 4 x 100	R\$ 110,00	2	R\$ 220,00
	Mão de obra	R\$ 600,00	1	R\$ 600,00
	Frete	R\$ 70,00	1	R\$ 70,00
	Software			
	Horas de desenvolvimento	R\$ 0,00	300	R\$ 0,00
	Eletrônica			
	Sensor VL53l0X (ML)	R\$ 40,49	3	R\$ 121,47
	Módulo GPS GY-NEO6MV2	R\$ 65,89	1	R\$ 65,89
	Servomotor	R\$ 13,50	1	R\$ 13,50
	Módulo Acelerômetro/Giroscópio MPU-6050	R\$ 14,90	1	R\$ 14,90
	Sensor VL53l0X (Ali)	R\$ 10,42	7	R\$ 72,94
	ATtiny85 (microcontrolador)	R\$ 13,92	10	R\$ 139,21
	Soquete Dip8 pinos	R\$ 1,04	10	R\$ 10,35
	Outros (Frete/IOF, etc)	R\$ 16,15	1	R\$ 16,15
	Energia			
	Dinamo	R\$ 150,00	1	R\$ 150,00
	Controlador de carga	R\$ 95,00	1	R\$ 95,00
	Bateria 12V 7Ah	R\$ 80,00	1	R\$ 80,00
			Total	3.203,05

Tabela 6 – Custos do projeto

4.4 Riscos

Após o andamento de 2 milestones, nossos riscos foram atualizados e a grande maioria teve um decréscimo no nível de risco ao projeto. Segue o diagrama de avanço dos riscos.

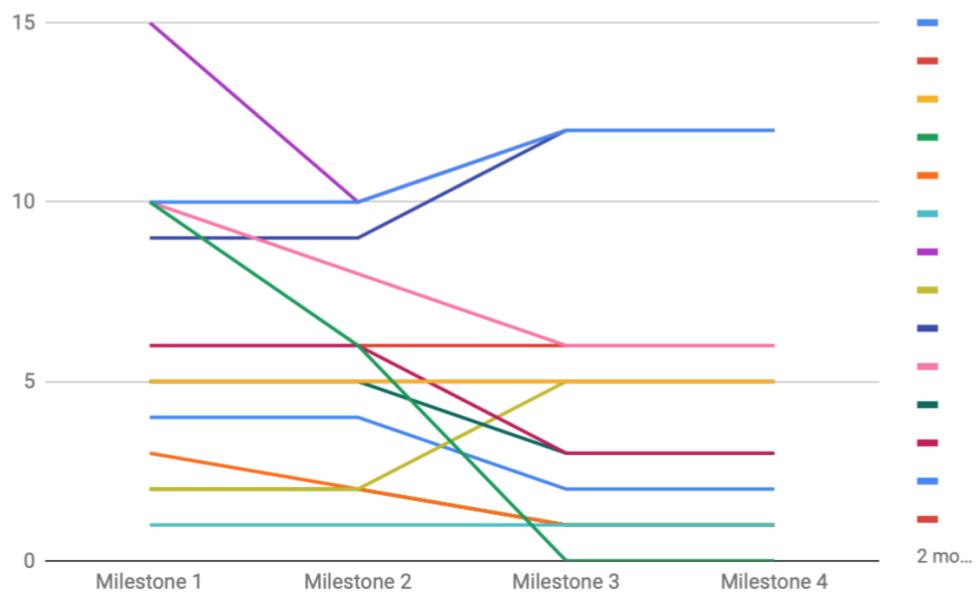


Figura 61 – Gráfico de riscos

Para visualizar todos os riscos do projeto, priorizados e com gráficos clique nesse [link](#)

Referências

APPLE. Core bluetooth programming guide. 2019. Disponível em: <https://developer.apple.com/library/archive/documentation/NetworkingInternetWeb/Conceptual/CoreBluetooth_concepts/AboutCoreBluetooth/Introduction.html>. Citado na página 68.

APPLE. Core location framework. 2019. Disponível em: <<https://developer.apple.com/documentation/corelocation>>. Citado na página 68.

APPLE. Swift. uma linguagem aberta e poderosa para todo mundo criar apps incríveis. 2019. Disponível em: <<https://www.apple.com/br/swift/>>. Citado na página 68.

APPLE. Xcode - apple developer. 2019. Disponível em: <<https://developer.apple.com/xcode/>>. Citado na página 68.

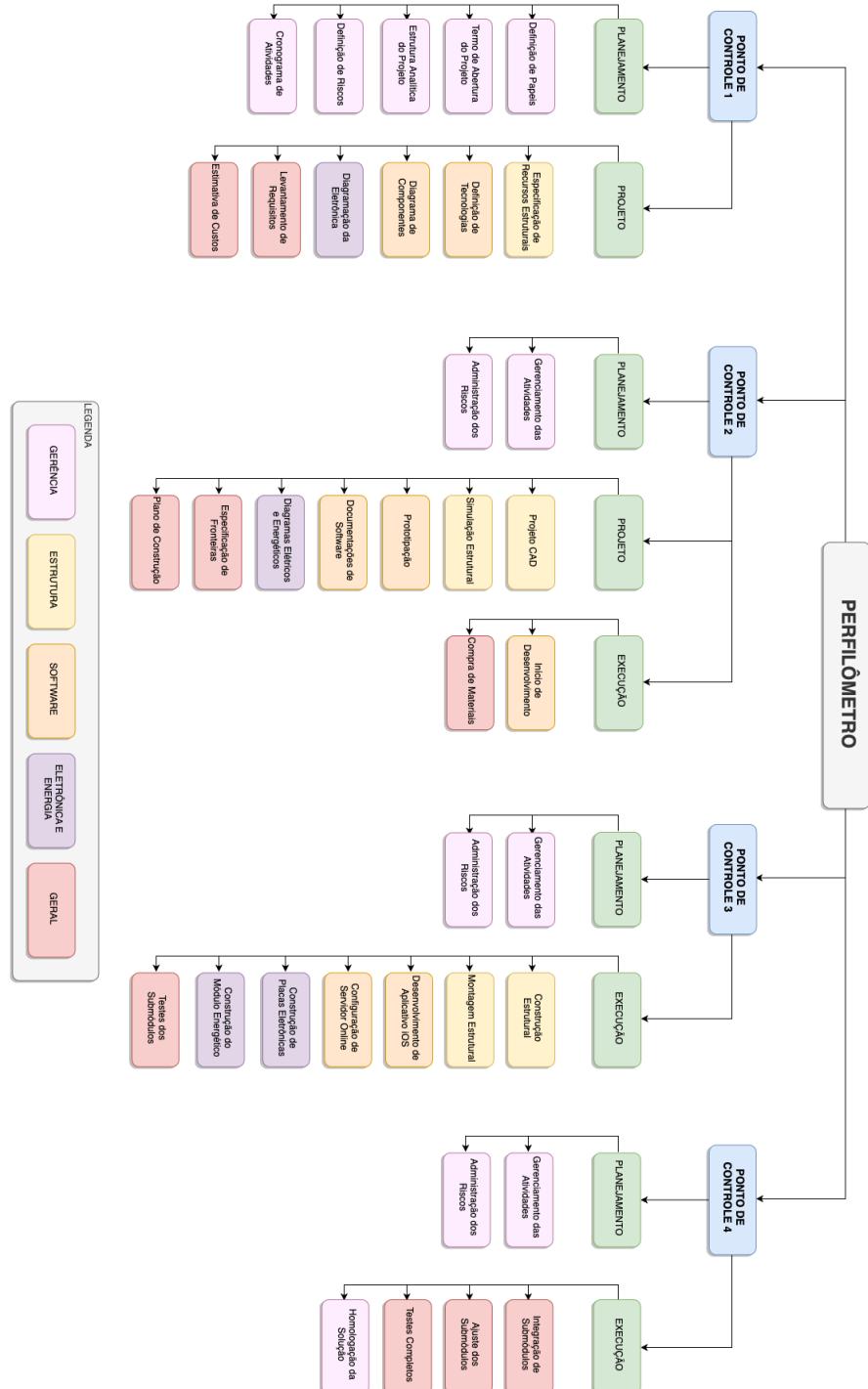
CABRAL, L. Introdução a comunicação spi. 2015. Disponível em: <<http://nerdeletreico.blogspot.com/2015/10/introducao-comunicacao-spi.html>>. Citado na página 31.

FIREBASE. Firebase helps mobile app teams succeed. 2019. Disponível em: <<https://firebase.google.com/?hl=pt-br>>. Citado na página 68.

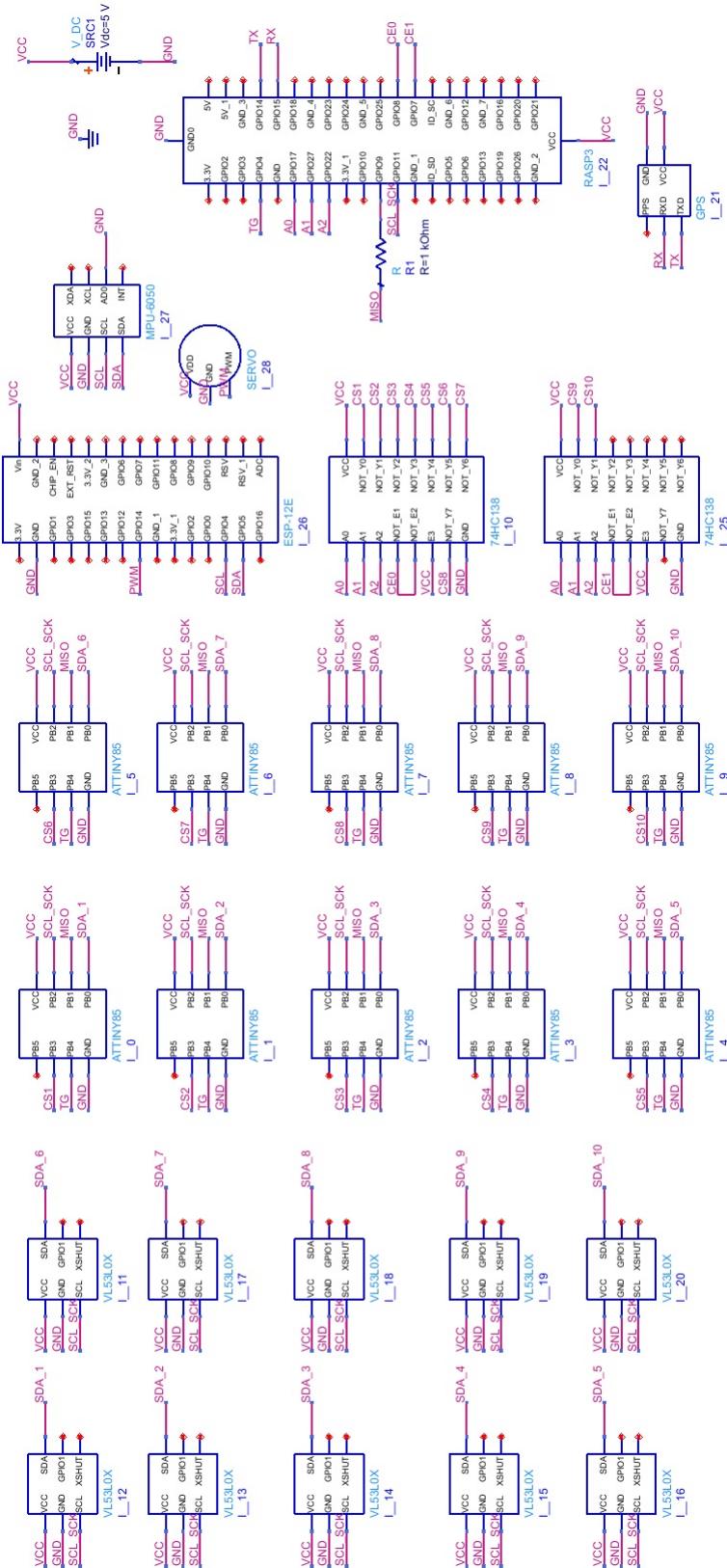
GINDI, D. C. Beautiful charts for ios/tvos/osx! the apple side of the crossplatform mpandroidchart. 2019. Disponível em: <<https://github.com/danielgindi/Charts>>. Citado na página 69.

GOOGLE. Maps sdk for ios. 2019. Disponível em: <<https://developers.google.com/maps/documentation/ios-sdk/intro?hl=pt-br>>. Citado na página 68.

ANEXO A - EAP Detalhada



ANEXO B – Esquemático Elétrico Geral



ANEXO C – Código da ESP-12E para teste do MPU

```
#include<Wire.h>
#include <Servo.h>

// Definindo o endereço I2C do MPU
int MPU = 0x68;
Servo myservo;

// Definindo os vetores de valores bruto das leituras do acelerometro e
// giroscopio
int16_t accResult[3], gyroResult[3];

// Definimos as taxas de conversão
#define A_R 16384.0 // mais ou menos 250 graus/seg
#define G_R 131.0 // mais ou menos 2 g

// Definimos a conversão de radianos para graus 180/PI
#define RAD_A_DEG = 57.295779

// Angulos
float Acc[2];
float Gy[3];
float Angle[2];

//Função de escrita em um dispositivo I2C
void writeTo(byte device, byte toAddress, byte val) {
    Wire.beginTransmission(device);
    Wire.write(toAddress);
    Wire.write(val);
    Wire.endTransmission();
}

//Função de leitura em um dispositivo I2C
void readFrom(byte device, byte fromAddress, int num, byte result[]) {
```

```
Wire.beginTransmission(device);
Wire.write(fromAddress);
Wire.endTransmission();
Wire.requestFrom((int)device, num);
int i = 0;
while(Wire.available()) {
    result[i] = Wire.read();
    i++;
}
}

//Função que lê as leituras do acelerômetro
void GetAccelerometerReadings(int16_t Result[]) {
    byte buffer[6];
    readFrom(MPU, 0x3B, 6, buffer);
    Result[0] = (((int16_t)buffer[0]) << 8) | buffer[1];
    Result[1] = (((int16_t)buffer[2]) << 8) | buffer[3];
    Result[2] = (((int16_t)buffer[4]) << 8) | buffer[5];
}

//Função que lê as leituras do giroscópio
void GetGyroscopeReadings(int16_t Result[]) {
    byte buffer[6];
    readFrom(MPU, 0x43, 6, buffer);
    Result[0] = (((int16_t)buffer[0]) << 8) | buffer[1];
    Result[1] = (((int16_t)buffer[2]) << 8) | buffer[3];
    Result[2] = (((int16_t)buffer[4]) << 8) | buffer[5];
}

void setup() {
    Wire.begin(D1,D2);
    //Wire.begin();
    writeTo(MPU, 0x6B, 0);
    Serial.begin(115200);

    myservo.attach(D5);
}

void loop() {
```

```
GetAccelerometerReadings(accResult);
GetGyroscopeReadings(gyroResult);

// Com os valores do acelerômetro é calculado os ângulos Y e X com a
fórmula da tangente.
Acc[1] = atan(-1*(accResult[0]/A_R)/sqrt(pow((accResult[1]/A_R),2) +
pow((accResult[2]/A_R),2)))*RAD_TO_DEG;
Acc[0] = atan((accResult[1]/A_R)/sqrt(pow((accResult[0]/A_R),2) +
pow((accResult[2]/A_R),2)))*RAD_TO_DEG;

// Calculando os ângulos do giroscópio
Gy[0] = gyroResult[0]/G_R;
Gy[1] = gyroResult[1]/G_R;
Gy[2] = gyroResult[2]/G_R;

// Aplicando o Filtro Complementar
Angle[0] = 0.98 *(Angle[0]+Gy[0]*0.010) + 0.02*Acc[0];
Angle[1] = 0.98 *(Angle[1]+Gy[1]*0.010) + 0.02*Acc[1];
Angle[2] = Angle[2] + Gy[2] ;/* 0.010;

// Mostrando os valores pelo monitor serial
Serial.print("Ângulo X: "); Serial.print(Angle[0]);
Serial.print("\n");
Serial.print("Ângulo Y: "); Serial.print(Angle[1]);
Serial.print("\n");
Serial.print("Ângulo Z: "); Serial.print(Angle[2]);
Serial.print("\n-----\n");

delay(10);

int angulo = 0;
angulo = map(Angle[0], -90, 90, 0, 180);

myservo.write(angulo);
}
```


ANEXO D – Código da ESP-12E para teste de memória

```
#include <FS.h>

int led = D1;
int chave = D0;

void writeFile(String state, String path) {
    File rFile = SPIFFS.open(path, "w+");
    if(!rFile){
        Serial.println("Erro ao abrir arquivo!");
    } else {
        rFile.println(state);
        Serial.print("gravou estado: ");
        Serial.println(state);
    }
    rFile.close();
}

String readFile(String path) {
    File rFile = SPIFFS.open(path, "r");
    if (!rFile) {
        Serial.println("Erro ao abrir arquivo!");
    }
    String content = rFile.readStringUntil('\r'); //desconsidera '\r\n'
    Serial.print("leitura de estado: ");
    Serial.println(content);
    rFile.close();
    return content;
}

void openFS(void){
    //Abre o sistema de arquivos
    if(!SPIFFS.begin()){
        Serial.println("\nErro ao abrir o sistema de arquivos");
    }
}
```

```
    } else {
        Serial.println("\nSistema de arquivos aberto com sucesso!");
    }
}

void setup() {
    pinMode(led, OUTPUT);
    pinMode(chave, INPUT);
    Serial.begin(9600);
    openFS();

    // no primeiro upload de programa o arquivo state.txt deve ser criado com o conteúdo
    // no segundo upload a linha deve ser comentada.
    //writeFile("off", "/state.txt");

    // verifica o último estado do LED e ativa de acordo
    String state = readFile("/state.txt");
    if(state == "on")
    {
        digitalWrite(led, HIGH);
    }
    else if(state == "off")
    {
        digitalWrite(led, LOW);
    }
}

void loop() {
    if(digitalRead(chave) == LOW)
    {
        String state = readFile("/state.txt");
        if(state == "off")
        {
            writeFile("on", "/state.txt");
            digitalWrite(led, HIGH);
        }
        else if(state == "on")
        {
            writeFile("off", "/state.txt");
        }
    }
}
```

```
    digitalWrite(led, LOW);
}
while(digitalRead(chave) == LOW);
}
}
```


ANEXO E – Código do Attiny85 para teste do *laser*

```
#include <Wire.h>
#include <VL53L0X.h>

VL53L0X sensor;

#define HIGH_SPEED

void setup() {
    Serial.begin(115200);

    Wire.begin();

    sensor.init();
    sensor.setTimeout(500);

#if defined LONG_RANGE

    // lower the return signal rate limit (default is 0.25 MCPS)
    sensor.setSignalRateLimit(0.1);

    // increase laser pulse periods (defaults are 14 and 10 PCLKs)
    sensor.setVcselPulsePeriod(VL53L0X::VcselPeriodPreRange, 18);
    sensor.setVcselPulsePeriod(VL53L0X::VcselPeriodFinalRange, 14);

#endif

#if defined HIGH_SPEED

    // reduce timing budget to 20 ms (default is about 33 ms)
    sensor.setMeasurementTimingBudget(20000);

#elif defined HIGH_ACCURACY
```

```
// increase timing budget to 200 ms
sensor.setMeasurementTimingBudget(200000);

#endif
}

void loop() {
    Serial.print(sensor.readRangeSingleMillimeters());

    if (sensor.timeoutOccurred()) { Serial.print(" TIMEOUT"); }

    Serial.println();
}
```