
**Universidade Federal de Ouro Preto**
**Instituto de Ciências Exatas e Aplicadas**
**Departamento de Computação e Sistemas**

# Uma Plataforma Base para Prototipação de Soluções em IoT Utilizando LoRa

**Marco Antônio de Oliveira Costa**

## TRABALHO DE CONCLUSÃO DE CURSO

ORIENTAÇÃO:
Prof. Vicente José Peixoto de Amorim

**Dezembro, 2022**
**João Monlevade–MG**

**Marco Antônio de Oliveira Costa**

# Uma Plataforma Base para Prototipação de Soluções em IoT Utilizando LoRa

Orientador: Prof. Vicente José Peixoto de Amorim

Monografia apresentada ao curso de Engenharia de Computação do Instituto de Ciências Exatas e Aplicadas, da Universidade Federal de Ouro Preto, como requisito parcial para aprovação na Disciplina "Trabalho de Conclusão de Curso II".

**Universidade Federal de Ouro Preto**

**João Monlevade**

**Dezembro de 2022**

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

C837p Costa, Marco Antonio de Oliveira .
Uma plataforma base para prototipação de soluções em IoT utilizando LoRa. [manuscrito] / Marco Antonio de Oliveira Costa. - 2021.
52 f.: il.: color., gráf., tab..

Orientador: Prof. Dr. Vicente José Peixoto de Amorim.
Monografia (Bacharelado). Universidade Federal de Ouro Preto. Instituto de Ciências Exatas e Aplicadas. Graduação em Engenharia de Computação .

1. Internet das coisas. 2. Engenharia de protótipos . 3. Engenharia de software. 4. Sistemas embarcados (Computadores). I. Amorim, Vicente José Peixoto de. II. Universidade Federal de Ouro Preto. III. Título.

CDU 004.41

Bibliotecário(a) Responsável: Flavia Reis - CRB6-2431

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E APLICADAS
DEPARTAMENTO DE COMPUTACAO E SISTEMAS

**FOLHA DE APROVAÇÃO**

**Marco Antônio de Oliveira Costa**

**Uma Plataforma Base para Prototipação de Soluções em IoT Utilizando LoRa**

Monografia apresentada ao Curso de Engenharia da Computação da Universidade Federal
de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Engenharia de Computação.

Aprovada em 20 de dezembro de 2021

Membros da banca

Doutor - Vicente José Peixoto de Amorim - Orientador (Universidade Federal de Ouro Preto)
Doutor - Filipe Nunes Ribeiro - (Universidade Federal de Ouro Preto)
Doutor - Welbert Alves Rodrigues - (Universidade Federal de Ouro Preto)

Vicente José Peixoto de Amorim, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 03/01/2022

Documento assinado eletronicamente por **Vicente Jose Peixoto de Amorim**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 04/01/2022, às 10:10, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0263830** e o código CRC **01C367EE**.

**Referência:** Caso responda este documento, indicar expressamente o Processo nº 23109.000083/2022-48 SEI nº 0263830

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35400-000
Telefone: - www.ufop.br

# Resumo

Equipamentos e objetos conectados à Internet que são capazes de monitorar e atuar em processos físicos, possuem um grande potencial de facilitar e melhorar a vida das pessoas além de aprimorar a eficiência de empresas. O conceito por trás desse tipo de tecnologia é chamado de Internet das Coisas e tem atraído cada vez mais a atenção mundial. Por ainda ser um campo com inúmeras possibilidades de melhorias, o tema é atrativo também para pesquisadores, desenvolvedores e empresários, que buscam desenvolver novas formas de atuar nas necessidades de mercado. Quando se trata de produtos que demandam por comunicações em longas distâncias e sem acesso à Internet em todos os pontos, ainda faltam soluções homologadas no Brasil que facilitem o seu processo de prototipação. Sendo assim, a proposta desse trabalho é criar uma plataforma capaz de facilitar a prototipação do hardware desse tipo de solução utilizando a tecnologia LoRa para comunicação. Apesar de possuir um custo um pouco maior, ela permite a troca de informações por longas distância com baixo consumo energético. Durante o decorrer do texto será possível observar que apesar das dificuldades enfrentadas, foi possível desenvolver uma ferramenta de custo relativamente baixo capaz de auxiliar na criação desses produtos inteligentes.

**Palavras-chaves**: internet das coisas. LoRa. sistemas embarcados. computação de borda.

# Abstract

Equipment and objects connected to the Internet that are capable of monitoring and acting on physical processes have a great potential to facilitate and improve people's lives and also to improve the efficiency of companies. The concept behind the type of technology is called the Internet of Things and it has attracted more and more attention worldwide. As it is still a field with possibilities for improvement, the topic is also attractive to researchers, developers and entrepreneurs, who seek to develop new ways of acting on market needs. When it comes to products that require communications over long distances and without Internet access at all nodes, there is still a lack of approved solutions in Brazil that facilitate your prototyping process. Thus, a proposal for this type of solution using LoRa technology for communication. Despite having a slightly higher cost, it allows the exchange of information over long distances with low energy consumption. During the course of the text, it will be possible to observe that despite the difficulties faced, it was possible to develop a relatively low-cost tool capable of creating intelligent products.

**Key-words**: internet of things. LoRa. edge computing. embedded systems.

# Lista de ilustrações


Figura 1 – Ilustração da arquitetura LoRaWAN. Fonte: Elaborado pelo autor. . . 17
Figura 2 – Ilustração do *Message Queuing Telemetry Transport* (MQTT). Fonte: Elaborado pelo autor. . . . . 18
Figura 3 – Solução do SensoTerra *Soil Moisture*. Fonte: (INSIGHTS, 2018). . . . . 20
Figura 4 – Vinduino *Starter Kit*. Fonte: (WORKS, 2018). . . . . 21
Figura 5 – Milesight EM500-SMTC. Fonte: (MILESIGHT, 2018). . . . . 21
Figura 6 – Módulo STM8L Discovery. Fonte: (STMICROELECTRONICS, 2019). 24
Figura 7 – Módulo LorRaMesh da Radioenge. Fonte: (RADIOENGE, 2019). . . . 25
Figura 8 – Software Termite. Fonte: (COMPUPHASE, 2019). . . . . 26
Figura 9 – Raspberry Pi Zero W. Fonte: (FOUNDATION, 2019c). . . . . 26
Figura 10 – Modelo da placa com as referências dos componentes. Fonte: Elaborado pelo autor. . . . . 30
Figura 11 – Modelo da placa com as referências dos componentes. Fonte: Elaborado pelo autor. . . . . 31
Figura 12 – Diagrama do circuito. Fonte: Elaborado pelo autor. . . . . 31
Figura 13 – Placa para teste do circuito principal. Fonte: Elaborado pelo autor. . . 32
Figura 14 – Versão final da placa. Fonte: Elaborado pelo autor. . . . . 32
Figura 15 – Arquitetura da aplicação. Fonte: Elaborado pelo autor. . . . . 34
Figura 16 – Dashboard da aplicação web. Fonte: Elaborado pelo autor. . . . . 37
Figura 17 – Informações do sensor selecionado. Fonte: Elaborado pelo autor. . . . . 38
Figura 18 – Ação de acionar porta. Fonte: Elaborado pelo autor. . . . . 39
Figura 19 – Gráficos com dados de umidade e temperatura de um módulo. Fonte: Elaborado pelo autor. . . . . 39
Figura 20 – Medição de corrente do cenário de teste 1. Fonte: Elaborado pelo autor. 40
Figura 21 – Medição de corrente do cenário de teste 1. Fonte: Elaborado pelo autor. 41
Figura 22 – Medição de corrente do cenário de teste 2. Fonte: Elaborado pelo autor. 42
Figura 23 – Medição de corrente do cenário de teste 2. Fonte: Elaborado pelo autor. 42
Figura 24 – Medição de corrente do cenário de teste 3. Fonte: Elaborado pelo autor. 43
Figura 25 – Medição de corrente do cenário de teste 3. Fonte: Elaborado pelo autor. 44

# Lista de tabelas

Tabela 1 – Formato dos pacotes. . . . . 29
Tabela 2 – Descrição das funções. . . . . 29
Tabela 3 – *Bill of Materials* (BOM) da placa desenvolvida. . . . . 31
Tabela 4 – Cenário de teste 1 (tensão - V) . . . . . 41
Tabela 5 – Cenário de teste 1 (corrente - mA) . . . . . 41
Tabela 6 – Cenário de teste 2 (tensão - V) . . . . . 42
Tabela 7 – Cenário de teste 2 (corrente - mA) . . . . . 43
Tabela 8 – Cenário de teste 3 (tensão - V) . . . . . 44
Tabela 9 – Cenário de teste 3 (corrente - mA) . . . . . 44

# Lista de abreviaturas e siglas

**Anatel** Agência Nacional de Telecomunicações

**CSS** *Chirp Spread Spectrum*

**IoT** *Internet of Things*

**LoRa** *Long Range*

**LPWAN** *Low-Power Wide-Area Network*

**IDE** *Integrated Development Environment*

**JSON** *JavaScript Object Notation*

**NoSQL** *Not Only SQL*

**M2M** *Machine to Machine*

**MVPs** *Minimum Viable Products*

**JWT** *JSON Web Token*

**PCB** *Printed Circuit Board*

**BOM** *Bill of Materials*

**MQTT** *Message Queuing Telemetry Transport*

**RTC** *Real Time Clock*

**EVM** *Evaluation Modules*

# Sumário


**1 INTRODUÇÃO . . . . . 12**
**1.1 Motivação e Problema . . . . . 12**
**1.2 Objetivos . . . . . 14**

**2 REVISÃO BIBLIOGRÁFICA . . . . . 15**
**2.1 Conceitos básicos . . . . . 15**
2.1.1 Internet das Coisas . . . . . 15
2.1.2 Sistemas embarcados . . . . . 15
2.1.3 Computação em nuvem . . . . . 16
2.1.4 Computação de borda . . . . . 16
2.1.5 LoRa . . . . . 17
2.1.6 MQTT . . . . . 18
2.1.7 NoSQL . . . . . 19
**2.2 Trabalhos Correlatos . . . . . 19**
2.2.1 Agricultura: Irrigação . . . . . 19
2.2.1.1 SensoTerra . . . . . 20
2.2.1.2 Vinduino . . . . . 21
2.2.1.3 Milesight IoT . . . . . 21

**3 DESENVOLVIMENTO . . . . . 23**
**3.1 Ferramentas e tecnologias . . . . . 23**
3.1.1 Git e GitHub . . . . . 23
3.1.2 Hardware . . . . . 23
3.1.2.1 KiCad . . . . . 23
3.1.2.2 Microcontrolador STM8L152C6T6 . . . . . 24
3.1.2.3 IAR Embedded Workbench . . . . . 25
3.1.2.4 Módulo LoRaMesh Radioenge . . . . . 25
3.1.2.5 Termite . . . . . 26
3.1.2.6 Raspberry . . . . . 26
3.1.3 Software . . . . . 27
3.1.3.1 Python . . . . . 27
3.1.3.2 JavaScript . . . . . 27
3.1.3.3 Node.js . . . . . 27
3.1.3.4 Heroku . . . . . 27
3.1.3.5 Mosca . . . . . 28
3.1.3.6 JSON . . . . . 28

3.1.3.7 React . . . 28
3.1.3.8 MongoDB . . . 28
3.1.3.9 Visual Studio Code . . . 28
**3.2 Metodologia . . . 28**
3.2.1 Hardware . . . 29
3.2.1.1 Biblioteca e Protocolo de Comunicação . . . 29
3.2.1.2 Placa Base . . . 30
3.2.1.3 Testes . . . 32
3.2.2 Software . . . 33
3.2.3 Sistema de irrigação . . . 33
3.2.3.1 Arquitetura da solução . . . 33
3.2.3.2 Gateway . . . 34
3.2.3.3 Broker MQTT . . . 34
3.2.3.4 Servidor de Aplicação . . . 35
3.2.3.5 Banco de Dados . . . 35
3.2.3.6 Aplicação Web . . . 36

**4 RESULTADOS . . . 37**
**4.1 Biblioteca e documentação . . . 37**
**4.2 Aplicações . . . 37**
4.2.1 Irrigação Inteligente . . . 37
**4.3 Consumo e alcance de transmissão do hardware . . . 40**
4.3.1 Consumo . . . 40
4.3.2 Alcance . . . 45

**5 CONCLUSÃO . . . 46**
**5.1 Trabalhos futuros . . . 46**
5.1.1 Consumo . . . 47
5.1.2 Alcance . . . 47

**REFERÊNCIAS . . . 48**

# 1 Introdução

O conceito de Internet das Coisas (*Internet of Things* (IoT)) pode ser abstraído como um paradigma em que vários objetos, chamados de nós, estão conectados à internet e comunicando entre si (GUBBI et al., 2012). Fomentado pela constante busca por diminuir perdas e custos, facilitar atividades, automatizar processos, aumentar a segurança e a qualidade de vida das pessoas (SUNDMAEKER et al., 2010), a IoT vem ganhando destaque em âmbitos cada vez mais abrangentes, tais como: cidades inteligentes, segurança, agricultura, inteligência artificial, entre outros. Esse termo foi estabelecido por Kevin Ashton em 1999 no contexto de gerenciamento de cadeia de suprimentos (ASHTON, 2009)

Quando o assunto é IoT, é inevitável pensar na escalabilidade da solução e suas dificuldades de implementação. Muito disso decorre da complexidade dos dados, custo de instalação e de banda, reparo dos equipamentos, segurança da aplicação e também do acesso à Internet. Para reduzir custos de hardware, é ideal que se tenha um conjunto de componentes e tecnologias que possua um equilíbrio entre ser confiável, acessível e possuir boa gestão de energia (SO et al., 2016). Inclusive, quando se trata de estudos envolvendo mobilidade ou áreas de difícil acesso, tanto o tamanho quanto o consumo de energia do equipamento envolvido pode prejudicar e até mesmo inviabilizar determinada solução ou pesquisa (AMELOOT; TORRE; ROGIER, 2018).

Em busca de suprir tais demandas, esse trabalho propõe o desenvolvimento de uma plataforma para a prototipação de soluções em IoT utilizando tecnologia *Long Range* (LoRa) para comunicação entre os nós. LoRa, é uma tecnologia *Low-Power Wide-Area Network* (LPWAN) capaz de transmitir dados por meio de ondas de rádio frequência. Essa tecnologia permite comunicações de até 15 quilômetros de distância em áreas abertas utilizando uma potência na ordem de 100mW (PETAJAJARVI et al., 2015).

A plataforma apresentada neste trabalho permitirá a construção de soluções que facilitam o monitoramento via sensores e a análise e intervenção através de atuadores. Mesmo separados por longas distâncias, os integrantes da rede poderão receber e enviar dados de/para um servidor *web*. Tais dados poderão ser analisados e tratados com um maior poder computacional disponível em nuvem, facilitando a implantação de inteligência automatizada aos processos.

## 1.1 Motivação e Problema

De acordo com a confederação nacional da indústria, com o crescimento do interesse acerca dos temas indústria 4.0 e, consequentemente, internet das coisas, empresas vêm

estudando cada vez mais o investimento nesse tipo de tecnologia (CNI, 2018). Esse cenário expõe uma oportunidade para desenvolvedores, engenheiros e pesquisadores criarem soluções nas mais diversas áreas que possam agregar inteligência a diferentes processos. Para isso, é comum que projetistas façam uso de plataformas de desenvolvimento na fase de prototipação e validação das ideias inicias, isso porque, durante o processo de prototipação, os requisitos e necessidades acabam mudando. E considerando que o custo de fabricação de um *Printed Circuit Board* (PCB)[1] no Brasil é extremamente alto comparado ao exterior, é inviável fabricar outra PCB sempre que houver a necessidade de mudar uma simples conexão.

Inicialmente, os projetistas costumam optar por realizar todo o processo em módulos de avaliação (*Evaluation Modules* (EVM)) e prototipação e somente depois de validado, embarcar os componentes eletrônicos necessários em uma PCB específica para o projeto. Inclusive, muitas vezes para projetos sem requisitos muito rígidos ou algo que não será escalável, é comum ver a solução final estar conectada de alguma forma em placas com propósito de prototipação.

Durante o desenvolvimento de produtos cujos processos e dispositivos são separados fisicamente por longas distâncias ou estão inseridos em uma área remota ou de difícil acesso, existem outras preocupações além do funcionamento do produto em si. Nesse tipo de cenário, é importante que o contato com o dispositivo embarcado seja feito com uma frequência reduzida, já que realizar esse acesso é uma operação custosa. Sendo assim, requisitos de eficiência energética e distância de comunicação são ainda mais rigorosos.

Além disso, também é importante que esse dispositivo possa se conectar com algum intermediário ou diretamente com a Internet. Dessa forma, é possível armazenar esses dados em nuvem e ainda emitir e receber ações via algum sistema remoto. Um bom exemplo de um cenário que pode ser aprimorado com o uso dessa tecnologia e que possui grandes áreas a serem cobertas, que muitas vezes são de difícil acesso, é a irrigação de uma grande lavoura de café.

Com a aplicação desse tipo de tecnologia nesse processo, é possível, por exemplo, encontrar quais os parâmetros agrários que geram uma safra de café de maior qualidade, monitorar todo o ambiente da fazenda e também atuar ativamente nele de qualquer lugar do mundo com acesso à Internet. Além disso, também é possível pensar em aplicações de inteligência artificial a fim de economizar parte da água da irrigação aproveitando as condições climáticas do local.

[1] Placa de circuito onde estão dispostos todos os componentes eletrônicos e suas ligações elétricas

## 1.2 Objetivos

O projeto tem como objetivo a construção de uma plataforma acessível de prototipagem de sistemas IoT com comunicação via protocolo LoRa. Utilizando um microcontrolador de baixo custo e baixo consumo, a plataforma visa permitir a comunicação entre sensores e atuadores por longas distâncias consumindo pouca energia. Facilitando a criação de *Minimum Viable Products* (MVPs) capazes de apresentar uma solução completa.

Também têm-se como objetivos específicos:

- Criar uma documentação de fácil acesso.
- Criar uma aplicação que comprove a funcionalidade da plataforma;
- Identificar os parâmetros de funcionamento da placa;
- Validação do sistema utilizando uma aplicação de irrigação inteligente; e
- Executar medições para estimar o gasto energético de uma solução real.

# 2 Revisão bibliográfica

Foram realizados estudos sobre as tecnologias de comunicação em longas distâncias, seus protocolos e suas aplicações em diversas áreas. Os levantamentos mais pertinentes ao contexto do trabalho são apresentados a seguir.

## 2.1 Conceitos básicos

Nessa seção serão apresentados conceitos básicos necessários para o entendimento completo da solução desenvolvida e que foram utilizados durante o desenvolvimento.

### 2.1.1 Internet das Coisas

(HALLER; KARNOUSKOS; SCHROTH, 2008) definem IoT como um mundo onde objetos físicos estão perfeitamente integrados em uma rede de informações, em que esses objetos podem ser participantes ativos em processos de negócio. Serviços estão disponíveis para interagir com esses objetos inteligentes através da Internet podendo requisitar quaisquer informações relacionadas a eles, sem abrir mão da segurança e da privacidade.

Já para (WEBER; WEBER, 2010), a Internet das Coisas é uma emergente arquitetura de informações baseada na Internet que facilita a troca de dados e serviços. A IoT tem o propósito de prover uma infraestrutura de tecnologia de informação que facilite a troca de dados de forma segura e confiável. Sua função é preencher a lacuna entre os objetos no espaço físico e sua representação nos sistemas de informação. Segundo os autores, a IoT irá servir para aumentar a transparência e realçar a eficiência das redes de cadeias de suprimentos.

Além disso, (WEBER; WEBER, 2010) estende o conceito criado por (HALLER; KARNOUSKOS; SCHROTH, 2008), citando que a IoT pode também servir como suporte principal para a computação onipresente, permitindo que ambientes inteligentes possam reconhecer e identificar objetos para buscar informações na Internet de forma a adaptar suas funcionalidades.

### 2.1.2 Sistemas embarcados

Segundo (HEATH, 2002), um sistema embarcado pode ser definido como um sistema baseado em microprocessador que é construído visando realizar uma função específica ou

um conjunto de funções. Além disso não é desenvolvido para ser programado pelo usuário final, o usuário apenas interage com o sistema, mas não altera sua funcionalidade.

Já para (KAELBLING, 1993), um sistema embarcado tem uma interação constante com um mundo externo dinâmico. Esses sistemas podem ser fisicamente incorporados ou podem ser puramente computacionais. Robôs móveis, controladores de processos industriais e programas de administrar agenda são exemplos de sistemas embarcados. É esperado que esses sistemas funcionem por longos períodos de tempo e que repetidamente processem novas entradas de dados e gerem dados de saída. Eles são embarcados em um ambiente cuja dinâmica pode influenciá-los mas não controlá-los completamente.

### 2.1.3 Computação em nuvem

Segundo (TAURION, 2009), o termo computação em nuvem, pode ser usado para descrever um ambiente de computação baseado em uma grande rede de servidores, virtuais ou físicos. De forma simplificada, é um conjunto de recursos com capacidade de processamento, armazenamento, conectividade, plataformas, aplicações e serviços disponibilizados na Internet.

Já segundo (FURHT, 2010), a computação em nuvem pode ser definida como um novo estilo de computação em que recursos dinamicamente escaláveis e frequentemente virtualizados são fornecidos como serviço pela Internet. Com o uso dessa tecnologia, os usuários podem acessar programas, base de dados, e aplicações através de diversos dispositivos que possuem acesso a Internet, tais como, computadores pessoais, notebooks e *smartphones*.

### 2.1.4 Computação de borda

Para (Singh, 2017), a computação em borda é um paradigma que traz a carga de trabalho e o processamento de dados mais próximos de onde eles serão utilizados, facilitando assim interações em tempo real. A computação em borda aos poucos vem migrando as aplicações, dados e serviços em nuvem de nós centralizados para as bordas da rede. Este fato permite que a análise e geração de conhecimento seja feita na fonte de dados ou o mais próximo a ela possível.

Já segundo (Shi et al., 2016), a computação em borda se refere a possibilitar que as tecnologias permitam realizar processamentos nas bordas da rede. Considera-se a borda como qualquer recurso de processamento ou rede ao longo do caminho entre a fonte de dados e o centro de processamento em nuvem.

Por exemplo, as televisões mais modernas normalmente possuem um sistema de assistente de voz. Caso todo o processamento da fala fosse realizado em nuvem levaria muito mais tempo para se obter as respostas, tanto por conta da velocidade do tráfego da

rede quanto por conta do próprio processamento. Sendo assim, parte desses tratamentos são feitos na borda da aplicação, ou seja, localmente no hardware do televisor.

### 2.1.5 LoRa

O termo LoRa, que é um acrônimo para *Long Range*, é uma tecnologia proprietária de modulação baseada em *Chirp Spread Spectrum* (CSS). Um pulso CSS é um sinal que varia sua frequência ao longo do tempo de transmissão. Além disso esses sinais se propagam pelas faixas de banda não licenciadas, o que legalmente facilita o seu uso.

A modulação CSS vem sido usado a muito tempo para comunicações em longa distância em aplicações militares e agências espaciais devido sua habilidade em resistir a interferências. LoRa é a primeira implementação a ser comercializada para infraestruturas de baixo custo que utiliza o CSS (KHUTSOANE; ISONG; ABU-MAHFOUZ, 2017).

Como a tecnologia LoRa é basicamente a forma de transmissão, é possível utilizar diferentes protocolos de comunicação e até mesmo criar o seu próprio. Apesar disso a LoRa Alliance (ALLIANCE, 2019) decidiu criar o LoRaWAN para padronizar essa comunicação. Um protocolo e sua arquitetura de rede tem uma grande influência em determinar o tempo de vida de bateria, capacidade de rede, qualidade de serviço, segurança e tipos de aplicações suportadas (KHUTSOANE; ISONG; ABU-MAHFOUZ, 2017). Dessa forma, com um protocolo único mantido pela própria empresa da tecnologia, é possível facilitar integrações entre sistemas e ainda garantir um padrão de qualidade em algumas variáveis.

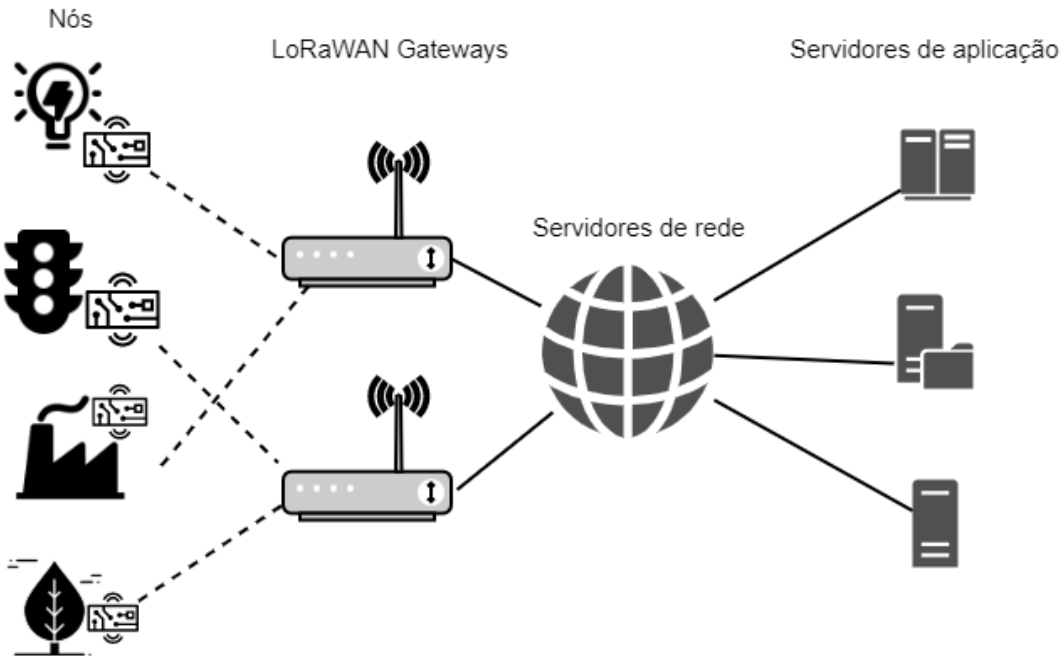



Figura 1 – Ilustração da arquitetura LoRaWAN. Fonte: Elaborado pelo autor.

Como pode ser visto na Figura 1, a arquitetura do LoRaWAN é descrita da seguinte

forma, existem os nós de coisas que podem ser quaisquer dispositivos capazes de receber ou enviar informações por meio do protocolo. Esses dispositivos se comunicam com o gateway, que é basicamente um dispositivo com acesso a rede capaz de se comunicar com o LoRaWAN. Este é responsável por receber e enviar informações para as coisas, e também por receber e repassar dados para a rede até os servidores de aplicação. Essa arquitetura é bem parecida com a que foi utilizada no desenvolvimento da aplicação descrita na Seção 3.2.3.

A possibilidade de transmissões em longas distâncias faz a tecnologia LoRa ser capaz de cobrir grandes terrenos. Além disso ainda possui um baixo consumo de energia, o que a torna perfeita para aplicações alimentadas por baterias. Para aplicações na agricultura isso é perfeito, já que remove a necessidade de fonte externa de alimentação no campo. (Davcev et al., 2018)

### 2.1.6 MQTT

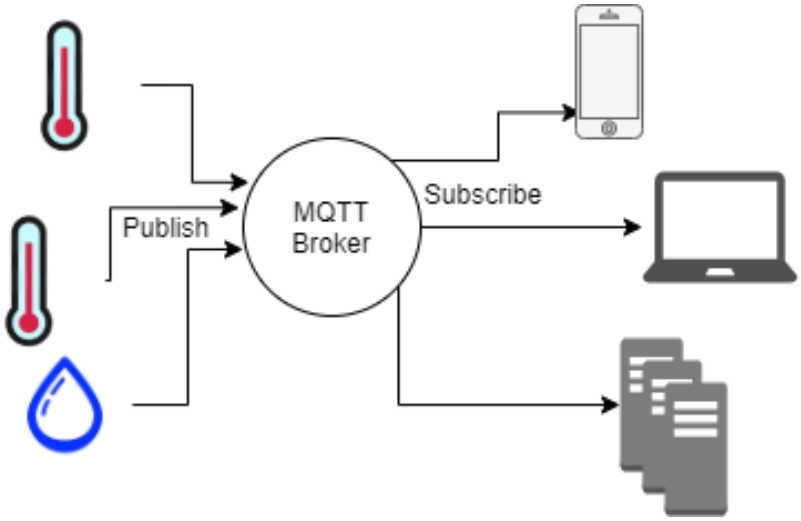



Figura 2 – Ilustração do MQTT. Fonte: Elaborado pelo autor.

O *Message Queuing Telemetry Transport* (MQTT) é um protocolo de transporte de mensagens fundamentado no modelo *publish/subscribe*. É um protocolo leve, aberto e desenvolvido para ser de fácil implementação. Essas características o fazem ser ideal para ser utilizado em diversas situações, tais como comunicação *Machine to Machine* (M2M) e contextos IoT onde a largura de banda é custosa para envio de vários pacotes pequenos.

O funcionamento do MQTT pode ser observado na Figura 2, consiste existe um *broker* que é o componente principal desse protocolo, responsável por fazer todo o gerenciamento das mensagens, recebendo e repassando os dados de/para o(s) clientes interessados.

É possível realizar a comunicação com o *broker* tanto por meio de *WebSockets* quanto puramente com o MQTT. Os navegadores ainda não possuem compatibilidade para realizar puramente com o MQTT. Tanto para receber quanto para enviar mensagens, basta instanciar um cliente MQTT, realizar a conexão com o *broker*, e fazer uma subscrição ou publicação em algum tópico determinado.

Esse tipo de funcionamento chamado de Pub/Sub, é muito utilizado em arquiteturas de microsserviços e em sistemas distribuídos em geral. Também disponibiliza toda a questão de autenticação de acesso, tanto ao *broker* quanto aos tópicos.

### 2.1.7 NoSQL

*Not Only SQL* (NoSQL) é um termo utilizado para representar bancos de dados que não possuem as tradicionais propriedades de bancos de dados relacionais. Esse tipo de modelagem utiliza diversos conceitos e técnicas que flexibilizam a modelagem de dados e a escalabilidade horizontal da aplicação, facilitando assim o seu uso para aplicações *Big Data* e *Internet of Things* (IoT). (Rautmare; Bhalerao, 2016)

## 2.2 Trabalhos Correlatos

No presente trabalho, busca-se o desenvolvimento de uma ferramenta que permita a prototipação de soluções capazes de realizar funções semelhantes às que serão aqui descritas. Soluções essas que possuam como necessidade de projeto possuir um baixo consumo energético e longo alcance.

### 2.2.1 Agricultura: Irrigação

Nessa seção serão apresentadas soluções correlatas ao trabalho corrente, com aplicações na área da agricultura e voltadas especificamente para a irrigação.

#### 2.2.1.1 SensoTerra

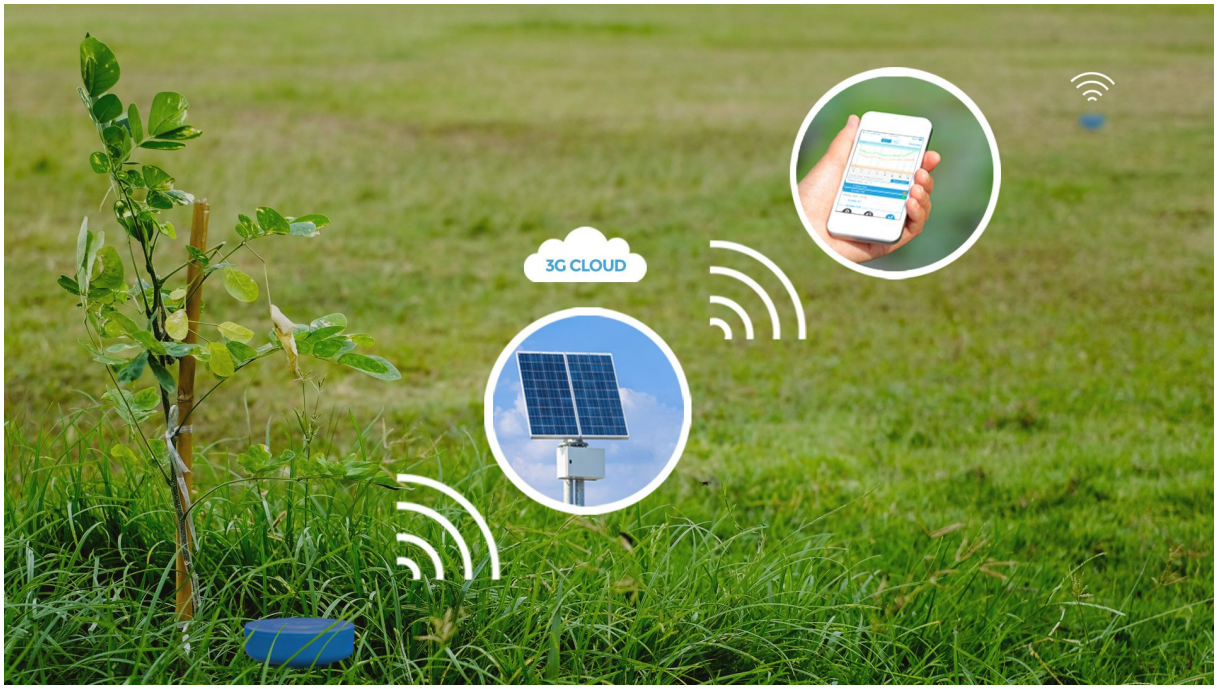



Figura 3 – Solução do SensoTerra *Soil Moisture*. Fonte: (INSIGHTS, 2018).

O SensoTerra (SENSOTERRA, 2019) possuí diversos dispositivos e plataformas voltadas para a agricultura e irrigação, um deles é o SensoTerra *Soil Moisture* que pode ser observado na Figura 3. Esta solução apresenta uma sonda de análise de solo que possui tecnologia LoRaWAN embarcada. A sonda foi construída para ser integrada facilmente a redes LoRaWAN, possuindo uma autonomia de bateria de 3 anos, compatibilidade com diversos tipos de solos e alcance de até 1,5 km. Essa solução possui um custo de aproximadamente 200 dólares por sonda e oferece um ambiente para visualização e análise dos dados.

#### 2.2.1.2 Vinduino


Figura 4 – Vinduino *Starter Kit*. Fonte: (WORKS, 2018).

Outro produto estudado foi o Vinduino (ELECTRONICDESIGN, 2017). Este possui um propósito parecido com o SensoTerra, de controle de irrigação, mas com uma construção diferente. O Vinduino é baseado em estações sensoriais capazes de acoplar até 4 sensores de umidade por estação. Além disso, possui um carregador solar de bateria e sua comunicação pode alcançar até 9 km. O custo do kit inicial, que pode ser visto na Figura 4 é aproximadamente 900 dólares e é composto por um *gateway*, uma estação e dois sensores de umidade. O produto também oferece uma plataforma de visualização e análise dos dados.

#### 2.2.1.3 Milesight IoT

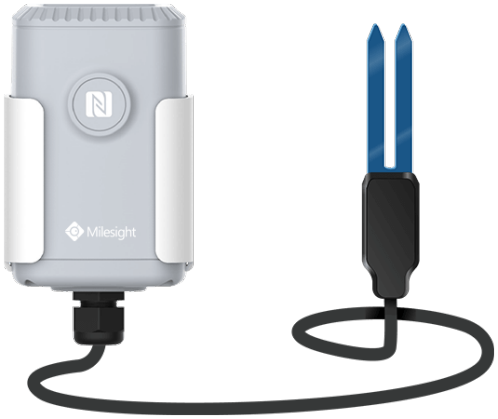



Figura 5 – Milesight EM500-SMTC. Fonte: (MILESIGHT, 2018).

Uma empresa que fornece produtos que possuem uma grande semelhança com a aplicação que será descrita na Seção 3.2.3 é a Milesight (MILESIGHT, 2021). A empresa

possui em seu portfólio os produtos Milesight UC51X de controle de válvulas solenoides e o Milesight EM500-SMTC que pode ser visto na Figura 5 de análise de solo. Ambos utilizam a tecnologia LoraWan e possuem uma longa vida útil de bateria de aproximadamente 10 anos, além de um alcance de transmissão de aproximadamente 5 km.

Além disso eles possuem uma interface de configuração via NFC e compatibilidade com o Milesight IoT Cloud (Web & App) que é a aplicação responsável pela integração dos dispositivos, visualização e controle dos sensores e também pela análise de dados.

O custo do controlador de válvula solenoide é de aproximadamente 300 dólares e o do sensor de análise de solo 600 dólares. A plataforma possui uma mensalidade baseada no número de dispositivos conectados, começando com um plano gratuito de até 5 nós e indo até 380 dólares para 500 dispositivos.

# 3 Desenvolvimento

Neste capítulo são apresentadas as etapas de construção da placa de circuito, da biblioteca para comunicação com o microcontrolador, as aplicações que utilizam a solução proposta e todas as ferramentas utilizadas no processo de desenvolvimento.

## 3.1 Ferramentas e tecnologias

Nessa seção serão apresentadas as ferramentas e tecnologias utilizadas separadas pelos contextos em que foram inseridas, para o desenvolvimento do hardware ou do software.

### 3.1.1 Git e GitHub

O Git (GIT, 2019) é uma ferramenta para controle de versão de código fonte muito usada para controlar fluxo de trabalho e gerir atividades em projetos de software, porém por ser uma ferramenta para controle de versão, também é possível utilizá-la para gerenciar diferentes versões de *design* de hardware, modelagens 3D, entre outros.

Já o GitHub (GITHUB, 2019), é uma plataforma para hospedagem de arquivos que utiliza o Git. Este foi utilizado para hospedar o código fonte e arquivos de todo o trabalho.

### 3.1.2 Hardware

Nessa seção serão descritas as ferramentas e tecnologias que foram utilizadas diretamente na construção da parte física do projeto.

#### 3.1.2.1 KiCad

O KiCad (KICAD, 2019) é uma ferramenta de código aberto para a criação de esquemas eletrônicos e *design* de placas de circuitos que incorpora em seu software ferramentas para gestão do projeto, editor de componente e de esquema, editor de placa de circuito e footprint, criação do modelo 3D dos circuitos além de um visualizador de arquivos Gerber. Por ser uma ferramenta relativamente simples, completa e de código aberto, ela foi utilizada para realizar o *design* da plataforma.

#### 3.1.2.2 Microcontrolador STM8L152C6T6

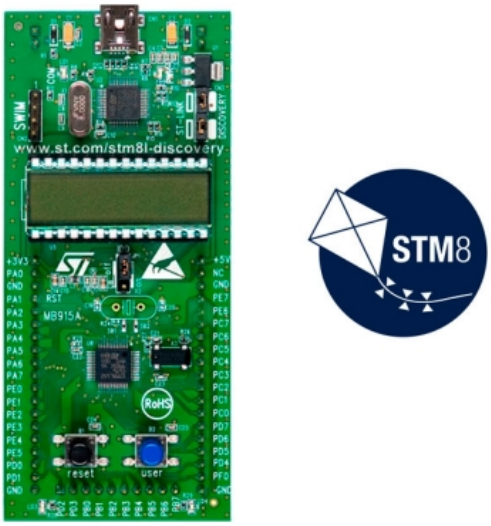



Figura 6 – Módulo STM8L Discovery. Fonte: (STMICROELECTRONICS, 2019).

O microcontrolador STM8L152C6T6 (STMICROELECTRONICS, 2019) é um microchip de 8 bits da empresa ST. Esse chip foi escolhido devido às suas funcionalidades voltadas para baixo consumo de energia, sendo elas:

- 5 modos de baixo consumo:
  - Modo de espera;
  - Modo baixa energia em funcionamento: 5,1 $\mu A$;
  - Modo baixa energia em espera: 3 $\mu A$;
  - Modo ativo-parado com *Real Time Clock* (RTC) ativo: 1.3 $\mu A$; e
  - Modo parado: 350 nA.
- Consumo: 195 $\mu A$/MHz + 440 $\mu A$.
- Rápida saída do estado de parado para um estado de funcionamento: 4.7 $\mu s$.

Além disso, existem diversas interrupções que podem ser programadas para transitar entre os estados de funcionamento do microcontrolador. Com relação às faixas de operação, esse chip trabalha de 1.8 V a 3.6 V e pode ser submetido à temperaturas de -40 °C a 125 °C.

Para realizar a prototipação, foi utilizado o módulo de *discovery* desse microcontrolador que pode ser visto na Figura 6.

#### 3.1.2.3 IAR Embedded Workbench

O IAR Embedded Workbench (SYSTEMS, 2019) é uma *Integrated Development Environment* (IDE) completa que possui suporte para o microcontrolador utilizado neste trabalho, como compilador, debugador, ferramentas de análises e controle de segurança. Essa ferramenta foi utilizada para programar o microcontrolador citado anteriormente.

#### 3.1.2.4 Módulo LoRaMesh Radioenge

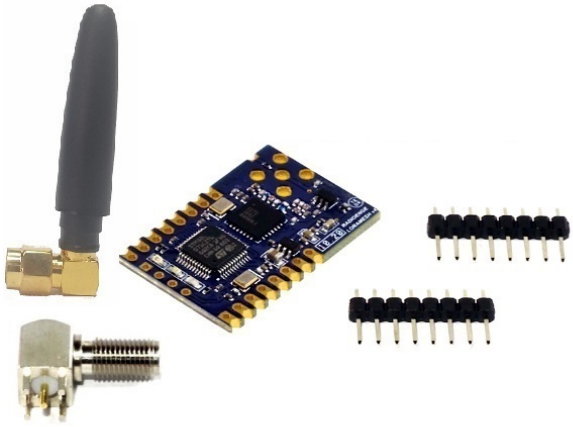


Figura 7 – Módulo LorRaMesh da Radioenge. Fonte: (RADIOENGE, 2019).

O módulo LoRaMesh (RADIOENGE, 2019) da Radioenge exibido na Figura 7, une a modulação LoRa à arquitetura de rede Mesh, em que cada rádio, além de ser um receptor de sinal, é também um roteador. Por possuir características que auxiliam na escalabilidade da aplicação e ser um módulo já homologado pela Agência Nacional de Telecomunicações (Anatel), ele foi utilizado neste trabalho.

#### 3.1.2.5 Termite

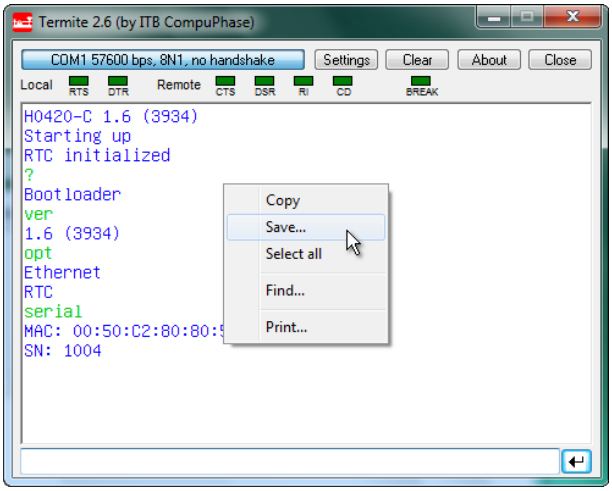



Figura 8 – Software Termite. Fonte: (COMPUPHASE, 2019).

Termite (COMPUPHASE, 2019) é um software de terminal RS232 de fácil uso e configuração. Ele oferece uma interface para acesso e comunicação com dispositivos via porta serial. Foi utilizado durante o desenvolvimento da biblioteca que comunica com o Módulo LoRaMesh Radioenge. Pode ser visto na Figura 8.

#### 3.1.2.6 Raspberry

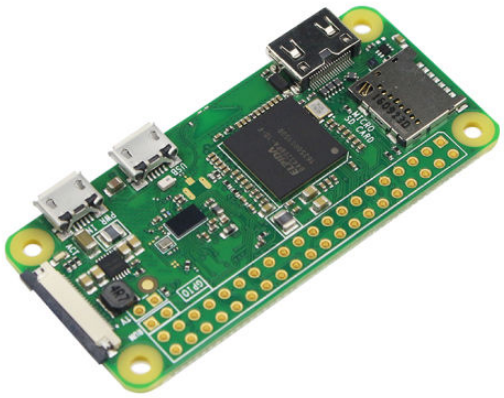


Figura 9 – Raspberry Pi Zero W. Fonte: (FOUNDATION, 2019c).

O Raspberry (FOUNDATION, 2019c) é uma série de microcomputadores desenvolvidos no Reino Unido para promover o ensino básico de ciência da computação. Hoje em dia com sua grande popularização ele é mundialmente usado em diversas áreas. Por

sua portabilidade, conectividade e poder de processamento, neste trabalho foi utilizado a versão Raspberry Pi Zero W, visto na Figura 9, para montar um *gateway* para a aplicação.

### 3.1.3 Software

Nessa seção serão descritas as ferramentas e tecnologias que foram utilizadas diretamente na construção da aplicação web, banco de dados e servidores do presente trabalho.

#### 3.1.3.1 Python

O Python (FOUNDATION, 2019b) é uma linguagem de programação interpretada, utilizada no desenvolvimento de *scripts* e robusta o suficiente para realizar grandes implementações. A linguagem possui diversas estruturas de dados já implementadas e uma grande coleção de módulos e pacotes desenvolvidos pela comunidade e disponibilizados através de um gerenciador de pacotes chamado pip. Foi utilizado para programar o hardware Raspberry Pi Zero W como um *gateway* da aplicação.

#### 3.1.3.2 JavaScript

O JavaScript (MOZILLA, 2019) é uma linguagem de programação interpretada com tipagem fraca, possui diferentes versões e é muito utilizada pela sua liberdade e rapidez no desenvolvimento, assim como o Python, a linguagem é extremamente robusta principalmente graças às atualização em seu core, visto que é uma linguagem que foi criada na década de 90. É a principal linguagem para programação de páginas web interativas e todos os principais navegadores tem um mecanismo para executá-la. A linguagem foi utilizada tanto no front-end quanto no back-end da aplicação.

#### 3.1.3.3 Node.js

O Node.js (FOUNDATION, 2019a) é um software capaz de executar o JavaScript fora de um navegador web. Foi desenvolvido para criar aplicações escaláveis e de baixo custo, sendo uma boa opção para implementação de microsserviços e componentes *serverless*, bem como para aplicações monolíticas. Foi utilizado para desenvolver o servidor de aplicação e o Broker MQTT citado na Seção 2.1.6.

#### 3.1.3.4 Heroku

O Heroku (SALESFORCE, 2019) é uma solução em nuvem de plataforma como serviço que permite a implantação de aplicações e servidores online, facilitando a gestão de infraestrutura e processos. Foi utilizado para hospedar o servidor da aplicação, a aplicação web, e o Broker MQTT.

#### 3.1.3.5 Mosca

Mosca (MCOLLINA, 2019) é um Broker MQTT que permite ser utilizado em aplicações que utilizam Node.js. Como o Heroku tem boa compatibilidade com o Node, o Mosca foi utilizado para o desenvolvimento desse *broker* da aplicação.

#### 3.1.3.6 JSON

O *JavaScript Object Notation* (JSON) (ORG, 2019) é um formato leve de dados que permite troca de informações simples e rápida entre sistemas. Possui uma semântica de fácil entendimento para as máquinas e legível a humanos, já que é no formato de chave e valor. Foi utilizado em todo processo de transferência de dados pela internet.

#### 3.1.3.7 React

O React (FACEBOOK, 2019) é uma biblioteca JavaScript para criar interfaces de usuário baseado em componentes. É muito utilizado mundialmente e possui muita documentação, por isso foi utilizado para o desenvolvimento da aplicação web neste trabalho.

#### 3.1.3.8 MongoDB

O MongoDB (MONGODB, 2019) é um banco de dados *Not Only SQL* (NoSQL) de código aberto, orientado a documentos e muito utilizado para o desenvolvimento de aplicações IoT. Este banco de dados oferece simplicidade, flexibilidade, versatilidade, segurança, escalabilidade, alta disponibilidade e diversas ferramentas úteis, como por exemplo a execução de funções de agregação em JavaScript durante as consultas. Neste trabalho, o MongoDB foi utilizando em conjunto com a ferramenta MongoDB Atlas, que permite a hospedagem em nuvem desse tipo de banco.

#### 3.1.3.9 Visual Studio Code

O Visual Studio Code (MICROSOFT, 2019) é um editor de código-fonte desenvolvido pela Microsoft. Inclui suporte para depuração, controle Git, além de diversas extensões que facilitam o desenvolvimento para uma grande quantidade de linguagens e frameworks. Foi utilizado durante o desenvolvimento do projeto.

## 3.2 Metodologia

Nessa seção são detalhados todos os passos seguidos para o desenvolvimento do presente projeto.

### 3.2.1 Hardware

Primeiramente será apresentado a parte metodológica voltado ao hardware desenvolvido.

#### 3.2.1.1 Biblioteca e Protocolo de Comunicação

Para realizar a comunicação do microcontrolador STM8L152C6T6 com o Módulo LoRaMesh Radioenge, foi necessário implementar uma biblioteca de comunicação serial compatível com este hardware.

Cada pacote de dados possui o formato descrito na Tabela 1, onde:

- **ID** (Bytes 0 e 1): Indica o identificador único do dispositivo dentro da rede.
- **Função** (Byte 2): Especifica o comando enviado ao dispositivo, sendo que a lista de comandos disponíveis estão descritas na Tabela 2.
- **Dado N** a **Dado 0** (Bytes 3 a N-1): Representam os parâmetros a serem enviados ou dados recebidos.
- **CRC** (Bytes N-1 e N): São os bytes para realizar a verificação de integridade do pacote.

Tabela 1 – Formato dos pacotes.

| **Byte 0** | **Byte 1** | **Byte 2** | **Byte 3** | | **Byte N-2** | **Byte N-1** | **Byte N** |
|---|---|---|---|---|---|---|---|
| ID(LSB) | ID(MSB) | Função | Dado N | ... | Dado 0 | CRC(LSB) | CRC(MSB) |

Tabela 2 – Descrição das funções.

| **Função** | **Descrição** |
|---|---|
| 0xD6 | Leitura / Escrita de parâmetros da modulação. |
| 0xE2 | Local Read - Leitura dos parâmetros do módulo conectado via UART. |
| 0xD4 | Remote Read - Leitura dos parâmetros de qualquer módulo na rede. |
| 0xCA | Write Config - Escrita dos parâmetros da rede no dispositivo (NET e ID). |
| 0xC2 | Comando I/O - Configura, lê e envia o estado dos pinos de GPIO. |
| 0xE7 | Diagnóstico - Adquire informações de operações do nó especificado. |
| 0xD8 | Ruído - Lê o nível de ruído no canal atual. |
| 0xD5 | RSSI - Lê as potências dos sinais entre o nó indicado e o nó vizinho. |
| 0xD2 | *Trace Route* - Traça a rota de comunicação do mestre a um nó específico. |
| 0x28 | Envia um pacote de dados para a interface serial transparente do destino. |
| <0x80 | Envia um pacote à interface de comandos do destino. |

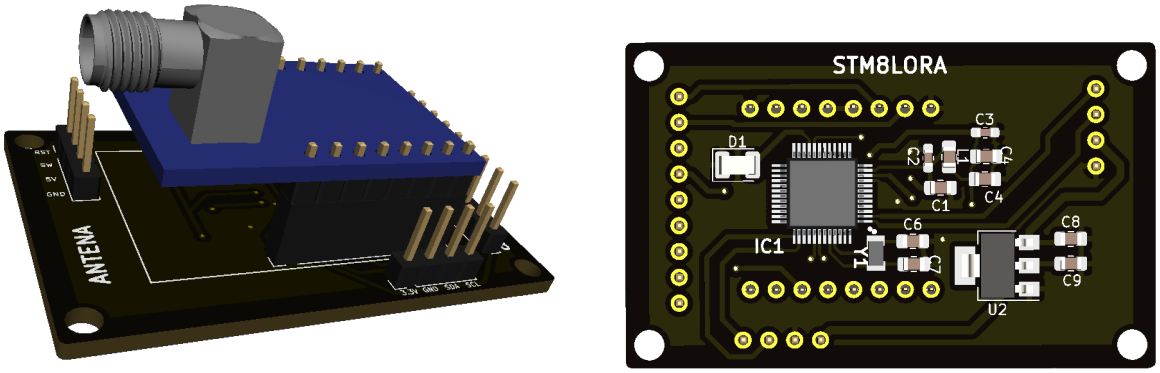
Para a criação da biblioteca, inicialmente foi realizado um estudo acerca do protocolo de comunicação utilizado pelo módulo a partir dos manuais fornecidos pelo fabricante.

Feito isso, foi realizada a codificação da biblioteca, que foi desenvolvida e testada utilizando a placa de desenvolvimento STM8L-Discovery (Kit de desenvolvimento inicial do microcontrolador STM8L152C6T6).

Apesar de possuir todas as funções descritas na Tabela 2, apenas as relacionadas à configuração do GPIO, leitura e escrita do GPIO e leitura dos parâmetros dos rádios foram disponibilizadas para o usuário.

Para a realização dos testes da biblioteca, foi criado um caso de uso simplificado para testar as comunicações e o funcionamento. Além disso, foi também assegurado, com o auxílio do software Termite, citado na Seção 3.1.2.5, que os dados corretos trafegavam pela interface serial.

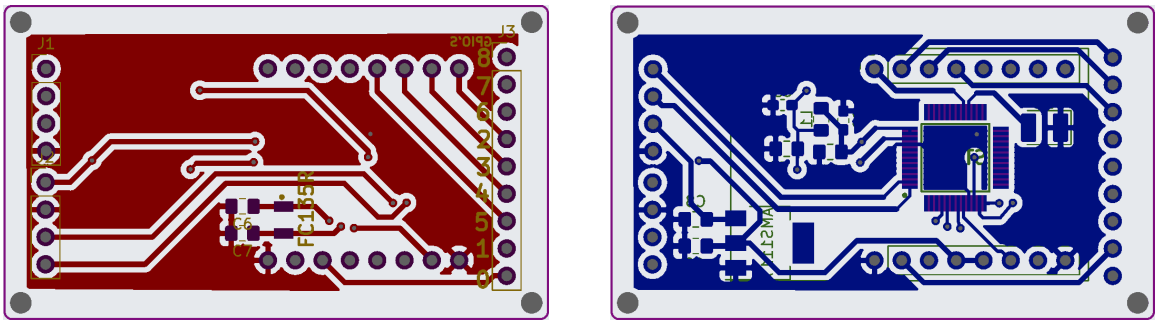
#### 3.2.1.2 Placa Base




(a) Vista lateral. (b) Face inferior.

Figura 10 – Modelo da placa com as referências dos componentes. Fonte: Elaborado pelo autor.

Para a modelagem da placa foi utilizado o software livre Kicad, citado em 3.1.2.1, em conjunto com o gerenciador de biblioteca de componentes Library Loader (SAMACSYS, 2019). Após a etapa de configuração inicial dos *softwares*, foi desenvolvido o esquemático do circuito, que pode ser visto na Figura 11(a) e na Figura 11(b), juntamente com o levantamento da *Bill of Materials* (BOM) apresentados na Tabela 3. Finalizado o circuito, que também está descrito pelo diagrama da Figura 12, foi feita a modelagem da placa de circuito impresso e também o modelo 3D para verificação de posicionamento dos componentes (Figura 14(a) e Figura 14(b)).



(a) Circuito elétrico face superior. (b) Circuito elétrico face inferior.

Figura 11 – Modelo da placa com as referências dos componentes. Fonte: Elaborado pelo autor.

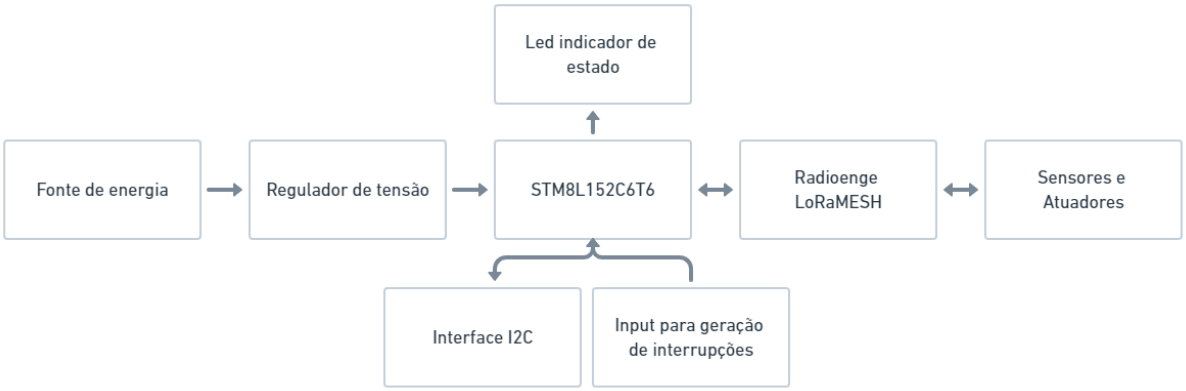



Figura 12 – Diagrama do circuito. Fonte: Elaborado pelo autor.

Tabela 3 – BOM da placa desenvolvida.

| Referência | Qtd | Dispositivo | Mfr.Part # | Preço Est.($) |
|---|---|---|---|---|
| IC1 | 1 | STM8L152C6T6 | STM8L152C6T6 | 1,78 |
| U1 | 1 | RadioEnge Lora | RD42C/RD49C | 16,2 |
| Y1 | 1 | FC-135R | FC-135R 32.7680KA-AC3 | 0,396 |
| C6,C7 | 2 | CAP CER 9PF | GRM0335C1H390JA01D | 0,01 |
| C1,C4,C5,C8 | 4 | CAP CER 0.1UF | GRM155R71E104KE14J | 0,06 |
| C2,C3 | 2 | CAP CER 1UF | CC0805MKY5V9BB105 | 1,7 |
| C9 | 1 | CAP CER 10UF | GRT31CR61H106ME01L | 0,2 |
| U2 | 1 | IC REG 3.3V | AMS1117-3.3 | 0,07 |
| D1 | 1 | LED WHITE | EAHE2835WD33 | 0,1 |
| L1 | 1 | FERRITE | HI0805R800R-10 | 0,04 |

Finalizada a modelagem, foi realizada a fabricação do circuito principal da PCB, que pode ser visualizada na Figura 13, manualmente utilizando os métodos convencionais de laboratório. Após atestar todo o funcionamento, foi realizado o pedido de fabricação em fábrica de algumas unidades da versão final. Posteriormente foi realizado a solda manual de todos os componentes e a versão final pode ser vista na Figura 14.

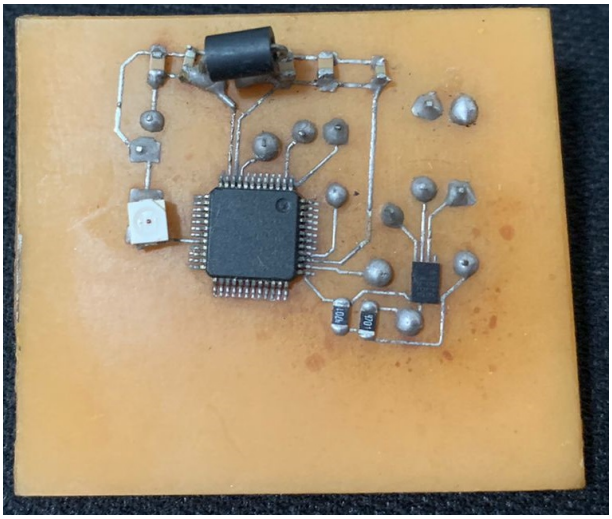
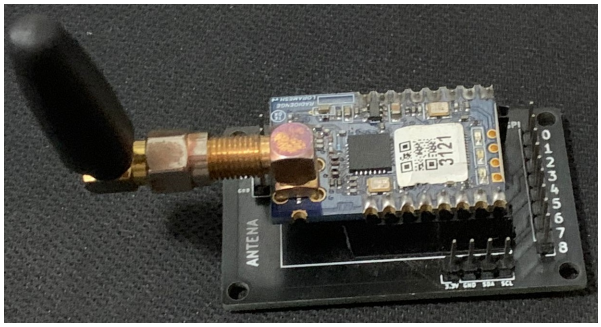
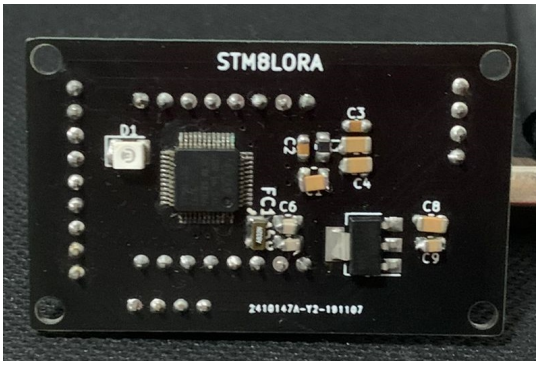


Figura 13 – Placa para teste do circuito principal. Fonte: Elaborado pelo autor.




(a) Vista lateral.




(b) Face inferior.

Figura 14 – Versão final da placa. Fonte: Elaborado pelo autor.

#### 3.2.1.3 Testes

Para aferir o consumo de energia da solução (rádio + placa) foram planejados os seguintes casos de teste.

1. Placa ligada por 5 minutos apenas aguardando por sinais.
2. Placa ligada por 5 minutos recebendo requisições de 1 em 1 minuto.
3. Placa ligada por 5 minutos recebendo requisições de 1 em 1 segundo.

Esses cenários de testes foram criados com o objetivo de cobrir todos, ou pelo menos a maioria, dos possíveis estados nos quais o hardware pode ser submetido.

Cada cenário de teste foi executado 5 vezes a fim de obter informações sobre o comportamento geral da plataforma.

Para realizar a aferição de qual seria o alcance médio de transmissão, foi criado uma aplicação simples capaz de enviar um sinal de liga e de desliga de um led. Dessa forma, o experimento foi conduzido da seguinte maneira. A cada 1 metro que o sensor se afastava do *gateway*, o mesmo enviava um sinal ao sensor, isso foi repetido até o ponto em que o sensor parou de responder ao *gateway*.

Por limitações de mobilidade, não foi possível realizar os testes em uma zona rural que possuísse uma área livre de interferências de construções maior do que o testado em zona urbana. Ou seja, não foi possível testar em zona rural uma distância que fosse maior do que o alcance máximo alcançado em zona urbana. Isso acaba impactando negativamente nos resultados já que a tendência é de que o alcance de comunicação aumentaria significantemente.

### 3.2.2 Software

Nessa seção será descrito o projeto na área de irrigação que utilizou a plataforma no seu desenvolvimento.

### 3.2.3 Sistema de irrigação

Para realizar uma prova de conceito foi desenvolvido pelo autor um sistema inteligente de irrigação conectado à internet, onde o usuário tem acesso às informações de estado de suas plantas (umidade e temperatura, via sensores), podendo influenciar manualmente no ambiente (via atuadores) ou utilizar o modo de execução automática do sistema.

#### 3.2.3.1 Arquitetura da solução

Como toda a parte de comunicação entre os rádios, aquisição de dados e atuação no meio já é provida pela plataforma criada, o foco foi então em desenvolver as outras camadas da aplicação.

Foi criado um *gateway* em um Raspberry Pi Zero W, citado na Seção 3.1.2.6, para encaminhar e receber os dados da Internet e também controlar os rádios da aplicação. Para realizar a comunicação com a Internet, foi utilizado o protocolo MQTT, citado na Seção 2.1.6.

O servidor de aplicação é responsável por enviar e receber dados do Broker MQTT, da aplicação web e do banco de dados.

A aplicação web por sua vez é a responsável por fornecer uma interface para visualização dos dados, bem como a manipulação do ambiente. A arquitetura segue ilustrada na Figura 15.

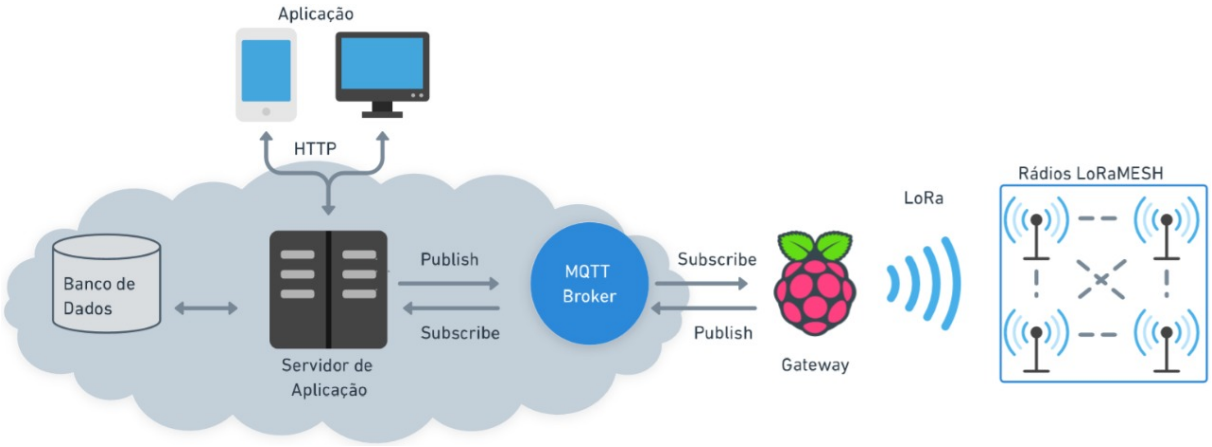



Figura 15 – Arquitetura da aplicação. Fonte: Elaborado pelo autor.

#### 3.2.3.2 Gateway

Como citado, o *gateway* é constituído por um Raspberry Pi Zero W, com o sistema operacional Raspberry Pi Lite, conectado a um Módulo LoRaMesh Radioenge. O software do *gateway* LoRa foi desenvolvido utilizando-se a linguagem Python, que facilita a integração com a internet por meio do protocolo MQTT. Para utilizar o Python foi necessária a implementação de uma biblioteca baseada na descrita na Seção 3.2.1.1.

Dessa forma, a execução do *gateway* se dá por um *script* em Python capaz de se comunicar com os módulos da rede mesh e com o Broker MQTT. A arquitetura da solução permite que parte do processamento seja feita no nós de borda da aplicação, tornando-a um pouco mais tolerante a falhas.

Para o *gateway* receber os dados do Broker MQTT é utilizado o seguinte padrão de objeto:

- **netId**: Id único de rede para referenciar um rádio dentro da rede.
- **port**: A porta do rádio que deve ser utilizada para realizar a ação.
- **action**: A ação que deve ser realizada: ler ou escrever.
- **value**: Em caso de escrita, esse campo indica o estado a ser escrito na porta, ligado ou desligado.

Já para enviar os dados para o servidor, é utilizado o mesmo formato do documento que será armazenado no banco, descrito na Seção 3.2.3.5.

#### 3.2.3.3 Broker MQTT

Como visto na Seção 2.1.6, o *broker* é uma peça fundamental no processo de transferência de dados por meio do protocolo MQTT e por isso foi desenvolvido de forma

a facilitar sua disponibilização na rede. Para fazer a disponibilização pelo Heroku, foi necessário utilizar a biblioteca Mosca, citada na seção 3.1.3.5, que é suportado de forma gratuita no servidor. Sendo assim, em seu desenvolvimento foi necessário realizar todas as configurações e políticas de rede manualmente.

#### 3.2.3.4 Servidor de Aplicação

O servidor de aplicação também foi desenvolvido utilizando o Node.js e hospedado no Heroku. As principais funções desse servidor são receber e enviar pacotes para o Broker MQTT, requisitar e armazenar dados no banco de dados e responder às requisições HTTP da aplicação web.

O servidor possui as seguintes rotas:

- **GET motes**: Retorna a listagem de todos os rádios de um determinado usuário.
- **POST motes**: Realiza a criação de um novo rádio.
- **GET infos**: Retorna informações gerais sobre o sistema de um determinado usuário.
- **GET data/:moteid**: Retorna os dados mais recentes de determinado rádio.
- **POST mqtt**: Realiza o envio de determinada ação para o gateway com o formato descrito na Seção 3.2.3.2.

Todas as rotas exigem autenticação que foi feita utilizando *JSON Web Token* (JWT).

#### 3.2.3.5 Banco de Dados

O banco de dados utilizado na aplicação foi o MongoDB. Para disponibilizá-lo na internet, foi utilizado o MongoDB Atlas que é uma ferramenta que permite a criação e hospedagem de clusters em nuvem. Para armazenar os dados dos rádios foi utilizado o seguinte formato de documento:

- **userId:** Id único para referenciar de qual usuário da aplicação o dado se trata.
- **moteNetId:** Id único de rede para referenciar um rádio dentro da rede.
- **sensors:** Um vetor contendo todos os sensores conectados ao rádio, cada posição do vetor contém um objeto com as seguintes informações:
  - **port:** A porta em que o sensor está conectado.
  - **value:** O valor retornado pelo sensor no formato:
    - **value:** O valor.

    - **type:** Tipo do valor. Ex: temperatura ambiente, umidade ambiente, umidade do solo, etc...

- **actuators:** Um vetor contendo todos os atuadores conectados ao rádio. Cada posição do vetor contém um objeto com as seguintes informações:
  - **port:** A porta em que o atuador está conectado.
  - **portSensorsRelated:** Um vetor indicando quais sensores estão relacionados a esse atuado, com indicação da porta do rádio e do id de rede.
  - **state:** O estado do atuador.
- **createdAt:** Horário em que o documento foi criado.

#### 3.2.3.6 Aplicação Web

Para permitir que um usuário final possa realizar interações com o sistema criado, visualizar facilmente os dados da lavoura e realmente ter um controle facilitado, à palma da mão, dos principais indicadores da sua fazenda, foi desenvolvido uma aplicação web.

Essa aplicação foi desenvolvida utilizando o React, citado na Seção 3.1.3.7, e nela o usuário é capaz de ver o posicionamento dos rádios em um mapa, visualizar os dados dos sensores, controlar os atuadores e o modo de execução.

Toda a interface foi pensada de forma a melhorar a experiência do usuário final com o sistema, além de sintetizar as informações mais importantes exibindo-as de forma clara e sucinta.

# 4 Resultados

Aqui serão descritos todos os resultados obtidos com o presente trabalho

## 4.1 Biblioteca e documentação

Como apresentado na Seção 3.2.1.1, foi desenvolvido uma biblioteca para realizar toda a comunicação serial entre os hardwares. Quando concluída, foi criada também uma documentação completa de uso e descrição das funcionalidades. Posteriormente, a documentação foi disponibilizada no GitHub (ANTONIO, 2021), visando permitir o acesso público, facilitar seu uso, além de permitir novas colaborações e usos em outros projetos.

Acessando o repositório também é possível fazer o download de todo o código fonte da biblioteca, cenários de usos e um código funcional de um exemplo prático. Durante o projeto ainda foram gerados insumos e casos de uso que foram publicados no blog do laboratório Imobilis. (IMOBILIS, 2019)

## 4.2 Aplicações

Aqui serão apresentados os resultados do desenvolvimento das aplicações.

### 4.2.1 Irrigação Inteligente

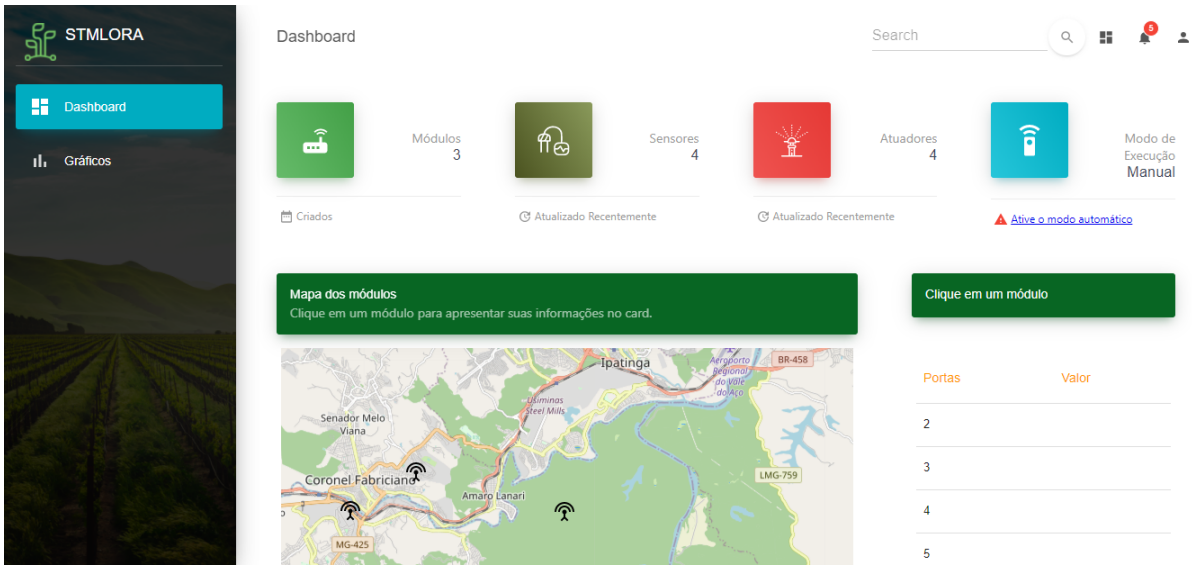



Figura 16 – Dashboard da aplicação web. Fonte: Elaborado pelo autor.

O *dashboard* desenvolvido para a aplicação web pode ser visto na Figura 16. Neste existem diversas informações e formas de controle dos módulos. As principais ações disponíveis são:

- Mudar o modo de funcionamento da aplicação:
  No canto superior existe o card exibindo qual o modo de funcionamento, podendo ser manual ou automático. No caso de manual, o sistema permite o acionamento dos módulos clicando em um dos módulos e escolhendo qual porta deseja acionar.

  No caso automático, fica a cargo do sistema determinar quando os atuadores serão acionados baseados nas informações dos sensores e dados da internet.

- Visualizar os dados dos módulos:
  Para visualizar os dados da última leitura dos sensores basta clicar em um dos módulos no mapa, e todas suas informações são exibidas na lateral direita.

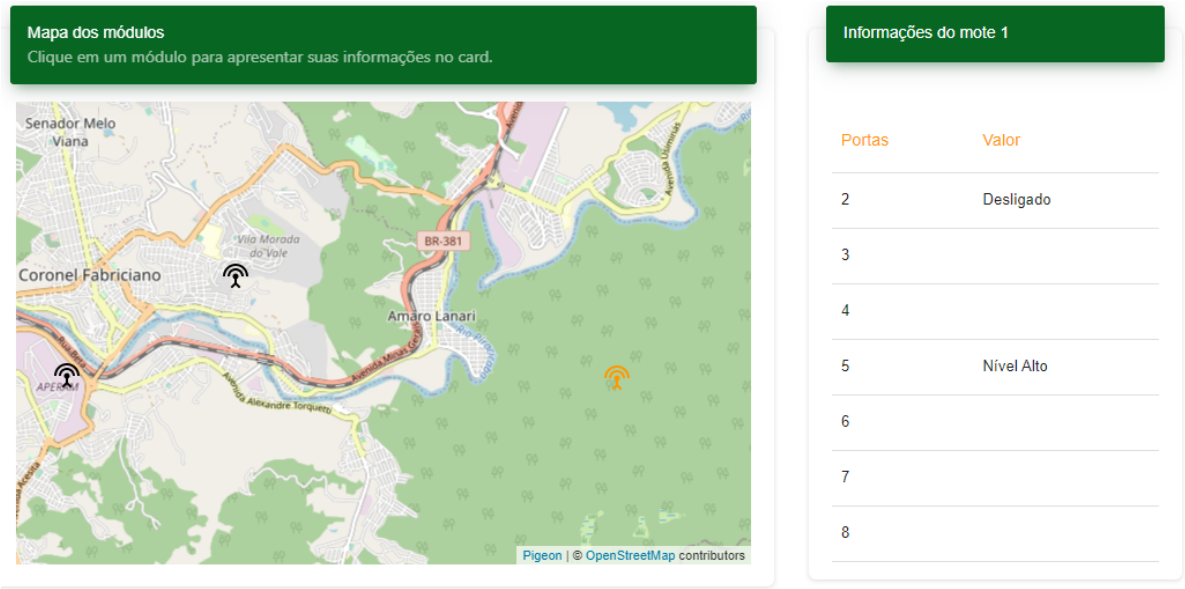



Figura 17 – Informações do sensor selecionado. Fonte: Elaborado pelo autor.

- Acionar uma porta:
  Para acionar uma porta, basta clicar no estado do pino que deseja ser ligado ou desligado que será exibido um modal, visto na Figura 18, para confirmar a ação.

  Caso confirmada, é enviado um POST para o servidor de aplicação que é responsável por fazer o tratamento dessa mensagem e repassar via MQTT para o *gateway*. Este enviará a mensagem para o módulo e então, caso confirmado, atualizará os dados de estado no banco de dados passando novamente pelo servidor.

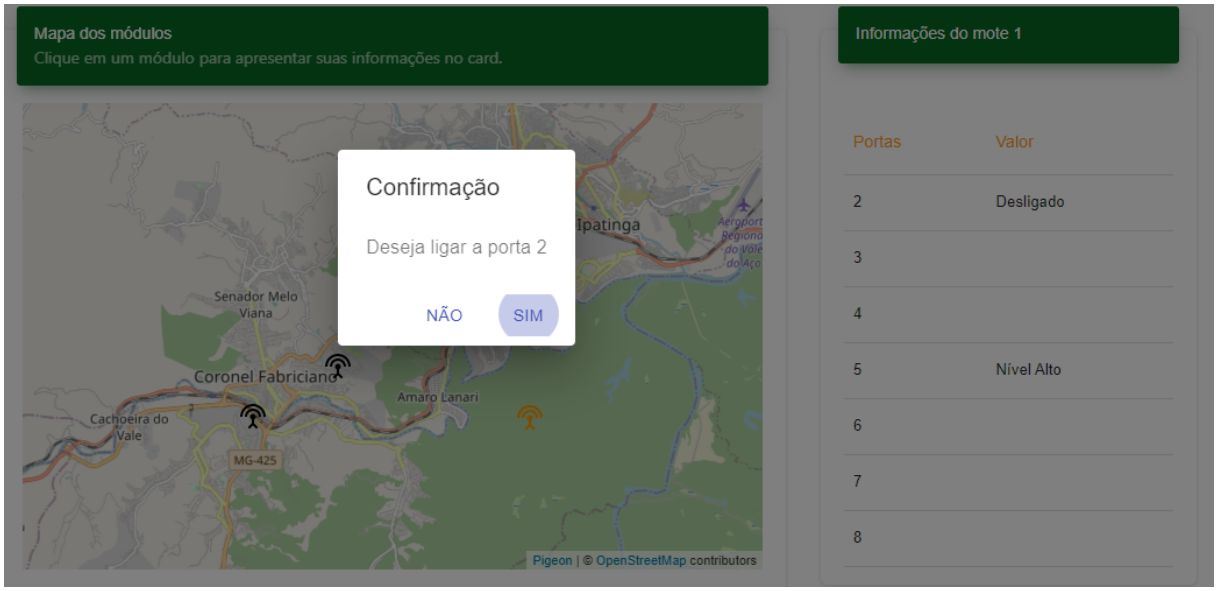



Figura 18 – Ação de acionar porta. Fonte: Elaborado pelo autor.

- Visualizar dados históricos:
  Também é possível exibir os dados históricos dos módulos no *dashboard* ou acessando a página de gráficos da aplicação.

  Como exibido na Figura 19, é possível visualizar os dados históricos dos módulos. Atualmente, para essa aplicação, os dados são exibidos separadamente com dois tipos diferentes, um para a umidade do solo, e um para a temperatura ambiente. Dessa forma, todos os sensores de um mesmo tipo são exibidos em conjunto em um mesmo gráfico.

  Ao selecionar um dispositivo no mapa, a aplicação faz a busca de todos os dados disponíveis do dia até o momento. Dessa forma, que todos os dados do *dashboard* se adaptam ao módulo selecionado. Também é possível selecionar um período de tempo qualquer para a visualização.

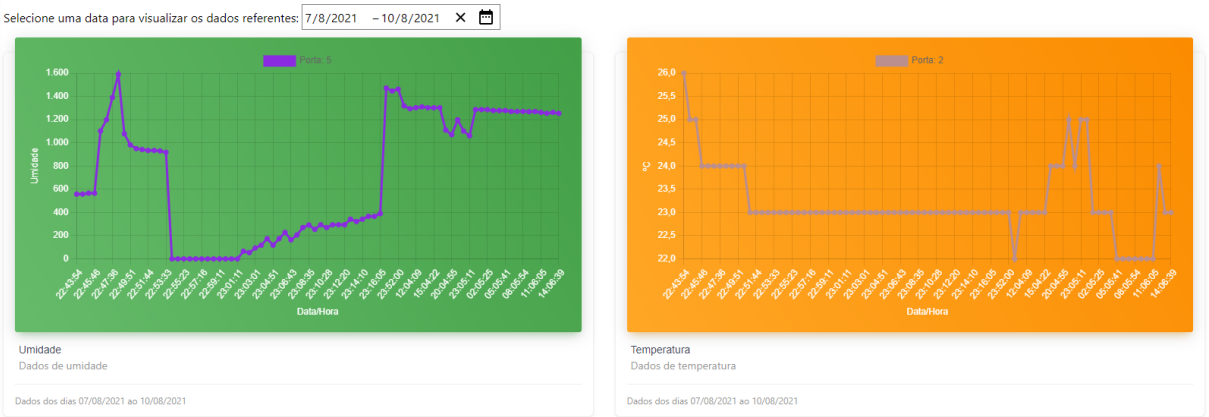



Figura 19 – Gráficos com dados de umidade e temperatura de um módulo. Fonte: Elaborado pelo autor.

## 4.3 Consumo e alcance de transmissão do hardware

Como apresentado na Seção 3.2.1.3, foram realizados testes para validação e verificação de consumo e alcance de transmissão da plataforma. Os resultados obtidos a partir das experimentações foram os seguintes:

### 4.3.1 Consumo

Foram realizadas três baterias de testes para verificação do consumo da placa, para cada um dos cenários obteve-se os seguintes dados:

- **Cenário 1**: Placa ligada por 5 minutos apenas aguardando por sinais:

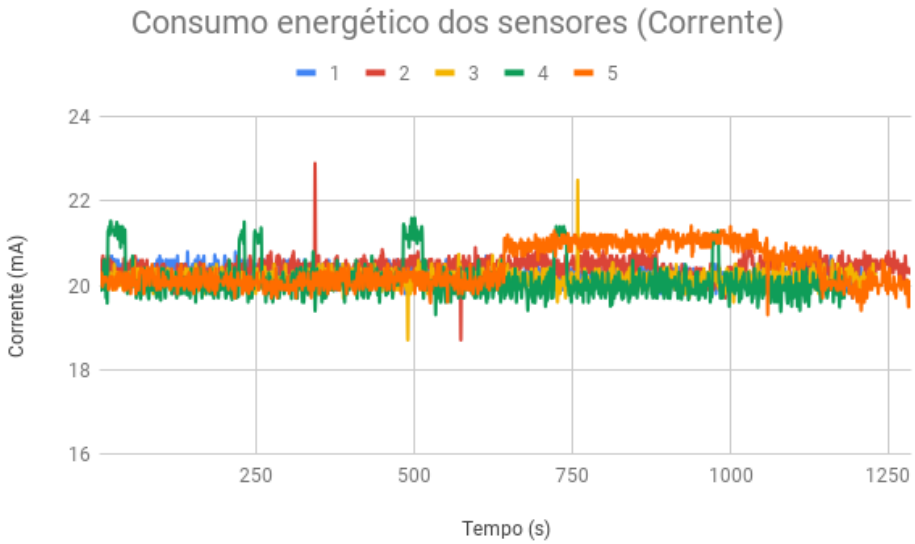



Figura 20 – Medição de corrente do cenário de teste 1. Fonte: Elaborado pelo autor.

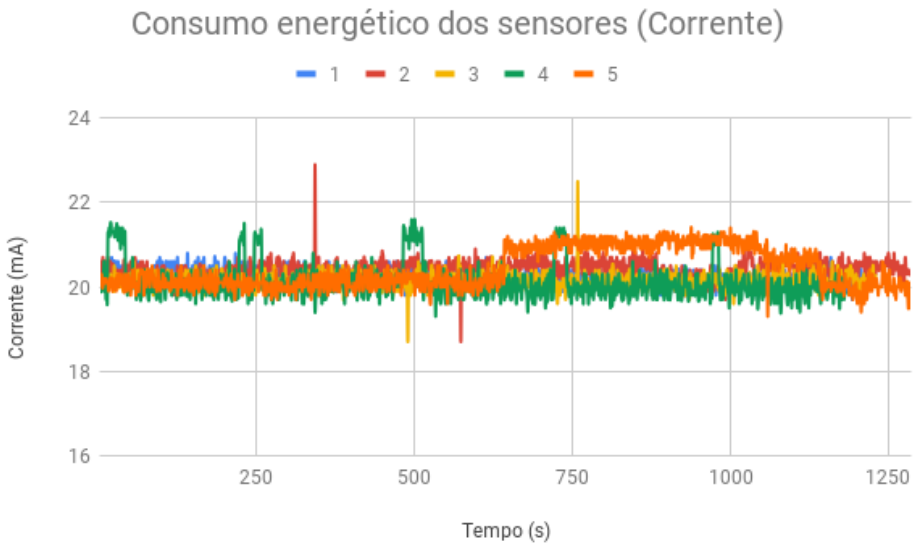



Figura 21 – Medição de corrente do cenário de teste 1. Fonte: Elaborado pelo autor.

Sendo assim, sumarizando os dados apresentados na Figura 20 e na Figura 21, temos os seguintes valores médios:

Tabela 4 – Cenário de teste 1 (tensão - V)

| Teste | Média (V) | Desvio Padrão |
|---|---|---|
| 1 | 2,039564541 | 0,02199674796 |
| 2 | 2,025375723 | 0,0177285325 |
| 3 | 2,013219512 | 0,01782087109 |
| 4 | 2,039625749 | 0,04886128707 |
| 5 | 2,01 | 0,04074269212 |

Tabela 5 – Cenário de teste 1 (corrente - mA)

| Teste | Média (mA) | Desvio Padrão |
|---|---|---|
| 1 | 20,39821151 | 0,222027948 |
| 2 | 20,25189769 | 0,1788707474 |
| 3 | 20,14365854 | 0,1804648563 |
| 4 | 20,4 | 0,4796767874 |
| 5 | 20,11 | 0,4118948458 |

Dessa forma, obteve-se um valor mínimo de tensão de 2,01 V e máximo de 2,04 V, que pode ser observado na Tabela 4. Já de corrente, foi medido um valor mínimo de 20,11 mA e máximo de 20,40 mA que pode ser observado na Tabela 5.

- **Cenário 2**: Placa ligada por 5 minutos recebendo requisições de 1 em 1 minuto.

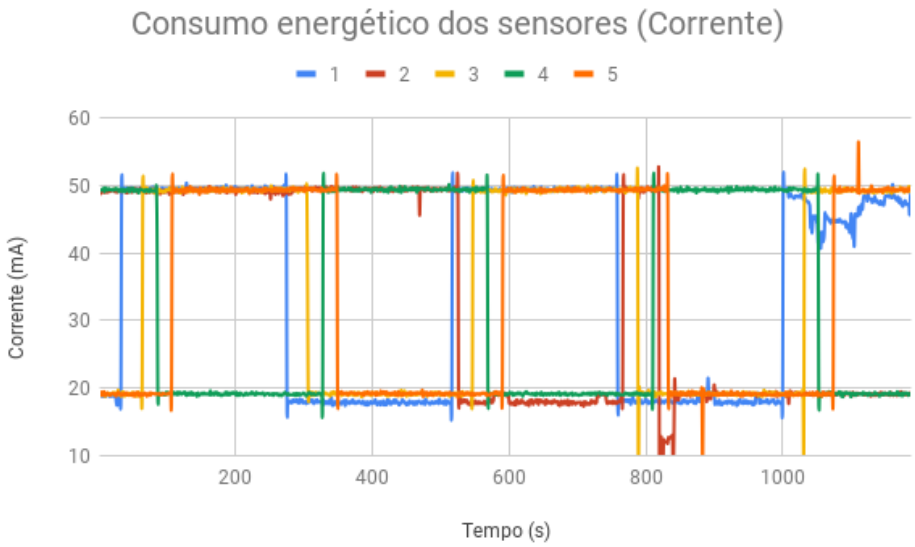



Figura 22 – Medição de corrente do cenário de teste 2. Fonte: Elaborado pelo autor.

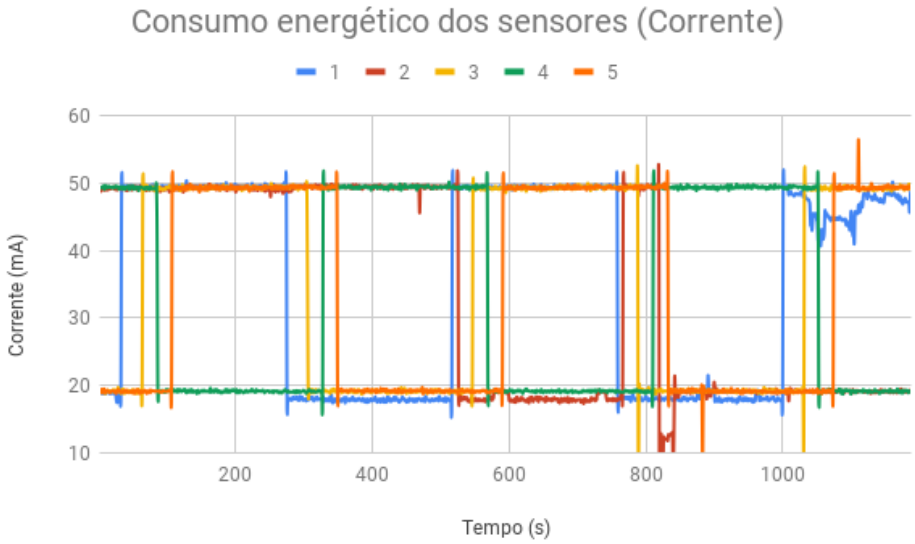



Figura 23 – Medição de corrente do cenário de teste 2. Fonte: Elaborado pelo autor.

Sendo assim, sumarizando os dados apresentados na Figura 22 e na Figura 23, temos os seguintes valores médios:

Tabela 6 – Cenário de teste 2 (tensão - V)

| **Teste** | **Média (V)** | **Desvio Padrão** |
|---|---|---|
| 1 | 3,55 | 1,521423232 |
| 2 | 3,345681818 | 1,551498161 |
| 3 | 3,54 | 1,50504494 |
| 4 | 3,361456229 | 1,512337741 |
| 5 | 3,436860269 | 1,51331757 |

Tabela 7 – Cenário de teste 2 (corrente - mA)

| Teste | Média (mA) | Desvio Padrão |
|---|---|---|
| 1 | 35,52 | 15,21543017 |
| 2 | 33,46607744 | 15,52118054 |
| 3 | 35,40 | 15,05987526 |
| 4 | 33,6037037 | 15,12217706 |
| 5 | 34,35968013 | 15,13642757 |

Dessa forma, obteve-se um valor mínimo de tensão de 3,34 V e máximo de 3,55 V, que pode ser observado na Tabela 6. Já de corrente, foi medido um valor mínimo de 33,46 mA e máximo de 35,52 mA que pode ser observado na Tabela 7.

- **Cenário 3**: Placa ligada por 5 minutos recebendo requisições de 1 em 1 segundo.

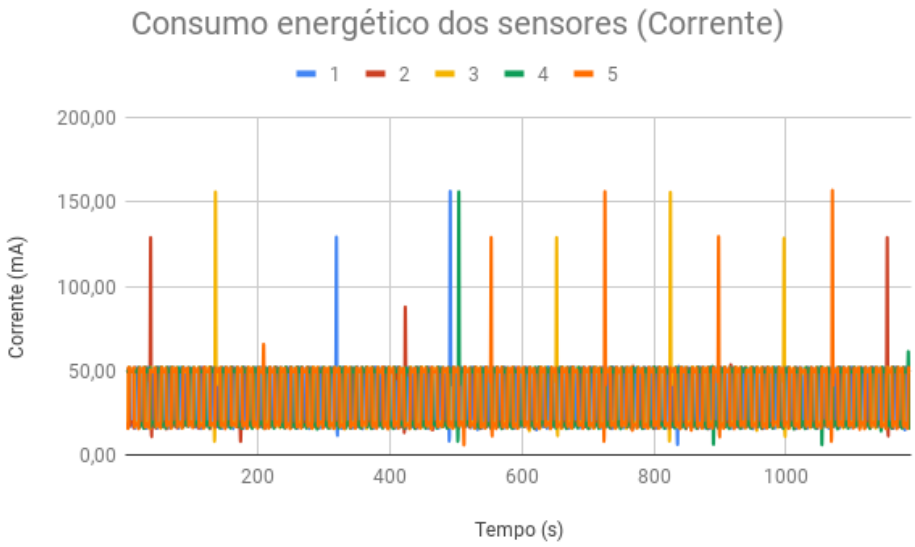



Figura 24 – Medição de corrente do cenário de teste 3. Fonte: Elaborado pelo autor.

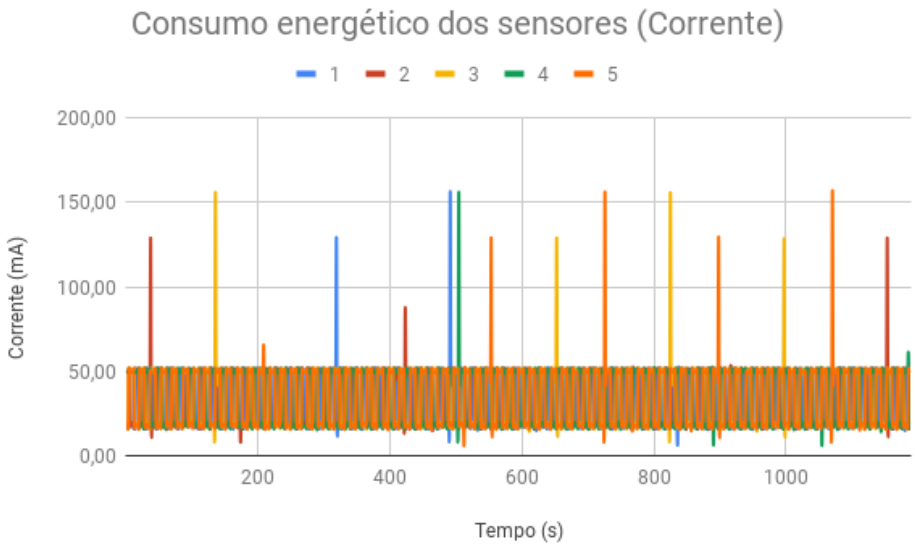



Figura 25 – Medição de corrente do cenário de teste 3. Fonte: Elaborado pelo autor.

Sendo assim, sumarizando os dados apresentados na Figura 24 e na Figura 25, temos os seguintes valores médios:

Tabela 8 – Cenário de teste 3 (tensão - V)

| **Teste** | **Média (V)** | **Desvio Padrão** |
|---|---|---|
| 1 | 3,43 | 1,600790875 |
| 2 | 3,43 | 1,565697887 |
| 3 | 3,42 | 1,606141953 |
| 4 | 3,45 | 1,663405455 |
| 5 | 3,42 | 1,635593171 |

Tabela 9 – Cenário de teste 3 (corrente - mA)

| **Teste** | **Média (mA)** | **Desvio Padrão** |
|---|---|---|
| 1 | 34,31 | 16,03257063 |
| 2 | 34,36 | 15,94399029 |
| 3 | 34,41 | 16,6292975 |
| 4 | 34,36 | 15,88712705 |
| 5 | 34,33 | 16,75063805 |

Dessa forma, obteve-se um valor mínimo de tensão de 3,42 V e máximo de 3,45 V, que pode ser observado na Tabela 8. Já de corrente, foi medido um valor mínimo de 34,31 mA e máximo de 34,41 mA que pode ser observado na Tabela 9.

Considerando então o caso mais leve de uso e o mais pesado, têm-se um consumo mínimo de aproximadamente 20,26 mA e um consumo máximo de 34,35 mA. Utilizando

duas células de bateria de notebook de 3400 mAh em paralelo, cada módulo de uma aplicação poderia ficar de 9 a 14 dias sem manutenção da fonte de energia.

### 4.3.2 Alcance

Para o alcance, foi possível realizar os testes de maior distância somente em área urbana, já que não foi possível acesso a uma área rural maior que o alcance máximo na área urbana, fato que prejudicou esse resultado. Notou-se que nas áreas de testes a interferência dos prédios foi muito grande por não ter sido possível posicionar o *gateway* em um ponto realmente superior aos prédios. Além disso foi utilizada uma antena simples de 2,15 dBi e 900 MHz tanto no *gateway* quanto nos módulos.

Com isso, o alcance do módulo foi de aproximadamente 290 metros de conexão direta em uma região com muitos prédios, podendo ser aumentado com a utilização de mais módulos de salto na rede mesh, além da troca da antena do *gateway*.

# 5 Conclusão

O presente estudo apresentou uma plataforma capaz de servir como base para a prototipação de dispositivos e sistemas IoT com comunicação via LoRa, bem como uma documentação da mesma, aplicações que já utilizam o hardware criado e testes de consumo e alcance do dispositivo.

É possível observar que é possível criar diferentes *Minimum Viable Products* (MVPs) de forma rápida e que já possuem a comunicação LoRa certificada pela Anatel. Sendo assim, podemos concluir que o objetivo deste trabalho foi atingido.

O desenvolvimento da aplicação utilizando a plataforma e dos testes realizados, além de atestarem seu funcionamento, também foi útil para gerar insumos de desenvolvimento de soluções em IoT e possíveis caminhos a se seguir utilizando a plataforma.

Sendo assim, analisando os objetivos específicos propostos no início do trabalho temos o seguinte cenário:

- Criar uma documentação bem definida: Foi concluída e está disponível na página do GitHub.
- Criar uma aplicação que comprove a funcionalidade da plataforma: Durante o presente trabalho foram apresentadas algumas das aplicações testadas para comprovar o funcionamento.
- Identificar os parâmetros de funcionamento da placa: Na etapa de testes foram apresentados os parâmetros de alcance e consumo da plataforma.
- Validação do sistema utilizando um caso real: Na Seção 4.2.1 é possível ver os resultados de uso em uma aplicação real.

Dessa forma, é possível constatar que os objetivos foram atingidos.

Apesar disso, comparando os trabalhos relacionados com a plataforma desenvolvida, é possível observar uma disparidade em consumo energético, alcance e produto final. Que se dá principalmente pela diferença no equipamento e protocolo utilizado.

## 5.1 Trabalhos futuros

Como observado nos resultados, ainda há espaço para melhorias na plataforma com relação ao alcance e consumo do hardware. Abaixo serão apresentadas sugestões de possíveis melhorias para trabalhos futuros.

### 5.1.1 Consumo

Para trabalhos futuros é possível realizar um mapeamento com mais faixas de utilização da plataforma visando traçar uma curva de consumo mais completa e facilitar a identificação de pontos de melhorias.

Além disso, também é possível remover o módulo regulador de tensão da plataforma para reduzir ainda mais o consumo. Tal modificação não foi realizada pela limitação de recursos do presente trabalho, visto que em caso de alguma falha ou acidente não seria possível obter novos módulos para continuação dos testes.

### 5.1.2 Alcance

Em relação ao alcance, é possível realizar testes com antenas de maior qualidade tanto para o *gateway* quanto para os módulos em si. A plataforma possui capacidade para antenas de 902MHz a 928MHz e 6dBi e foi utilizado uma antena com 2dBi na mesma faixa de frequência.

Além disso, também é possível fazer um comparativo financeiro para descobrir o *tradeoff* entre alcance e consumo.

# Referências


ALLIANCE, L. *LoRa Alliance*. 2019. Disponível em: <https://lora-alliance.org/>. Acessado em: 2019-08-13. Citado na página 17.

AMELOOT, T.; TORRE, P. V.; ROGIER, H. A compact low-power lora iot sensor node with extended dynamic range for channel measurements. *Sensors*, v. 18, p. 2137, 07 2018. Citado na página 12.

ANTONIO, M. *LoRaMESH STM8L Library*. 2021. Disponível em: <https://github.com/MarcoAOC/LoRaMESH_STM8L>. Acessado em: 2021-07-07. Citado na página 37.

ASHTON, K. That 'internet of things' thing. *RFiD Journal*, v. 22, p. 97–114, 01 2009. Citado na página 12.

CNI. *Investimentos em Indústria 4.0*. 2018. Disponível em: <https://www.portaldaindustria.com.br/estatisticas/pqt-investimentos-em-industria-40/>. Acessado em: 2019-08-13. Citado na página 13.

COMPUPHASE. *Termite: a simple RS232 terminal*. 2019. Disponível em: <https://www.compuphase.com/software_termite.htm>. Acessado em: 2019-08-13. Citado 2 vezes nas páginas 6 e 26.

Davcev, D. et al. Iot agriculture system based on lorawan. In: *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*. [S.l.: s.n.], 2018. p. 1–4. ISSN null. Citado na página 18.

ELECTRONICDESIGN. *Vinduino: An Open-Source, Affordable Water-Saving Technology*. 2017. Disponível em: <https://www.electronicdesign.com/industrial/vinduino-open-source-affordable-water-saving-technology>. Acessado em: 2019-08-13. Citado na página 21.

FACEBOOK. *React: Uma biblioteca JavaScript para criar interfaces de usuário*. 2019. Disponível em: <https://pt-br.reactjs.org/>. Acessado em: 2019-08-13. Citado na página 28.

FOUNDATION, O. *Node.Js*. 2019. Disponível em: <https://nodejs.org/en/>. Acessado em: 2019-08-13. Citado na página 27.

FOUNDATION, P. S. *Python.org*. 2019. Disponível em: <https://www.python.org/>. Acessado em: 2019-08-13. Citado na página 27.

FOUNDATION, R. P. *Teach, Learn, and Make with Raspberry Pi*. 2019. Disponível em: <https://www.raspberrypi.org/>. Acessado em: 2019-08-13. Citado 2 vezes nas páginas 6 e 26.

FURHT, B. Cloud computing fundamentals. In: _____. *Handbook of Cloud Computing*. Boston, MA: Springer US, 2010. p. 3–19. ISBN 978-1-4419-6524-0. Disponível em: <https://doi.org/10.1007/978-1-4419-6524-0_1>. Citado na página 16.

GIT. *Git*. 2019. Disponível em: <https://git-scm.com/>. Acessado em: 2019-08-13. Citado na página 23.

GITHUB. *Where the world builds software*. 2019. Disponível em: <https://git-scm.com/>. Acessado em: 2019-08-13. Citado na página 23.

GUBBI, J. et al. Internet of things (iot): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, v. 29, 07 2012. Citado na página 12.

HALLER, S.; KARNOUSKOS, S.; SCHROTH, C. The internet of things in an enterprise context. In: . [S.l.: s.n.], 2008. )5468, p. 14–28. ISBN 978-3-642-00984-6. Citado na página 15.

HEATH, S. *Embedded Systems Design*. Elsevier Science, 2002. ISBN 9780080477565. Disponível em: <https://books.google.com.br/books?id=BjNZXwH7HlkC>. Citado na página 15.

IMOBILIS, L. *Laboratório iMobilis*. 2019. Disponível em: <http://www2.decom.ufop.br/imobilis/>. Acessado em: 2019-08-13. Citado na página 37.

INSIGHTS, E. I. *Sensoterra chooses Senet LPWA network for soil sensors*. 2018. Disponível em: <https://enterpriseiotinsights.com/20170725/news/sensoterra-soil-sensors-tag4>. Accessado em: 2019-05-26. Citado 2 vezes nas páginas 6 e 20.

KAELBLING, L. *Learning in Embedded Systems*. MIT Press, 1993. (A Bradford book). ISBN 9780262111744. Disponível em: <https://books.google.com.br/books?id=WiN53ZYd0kgC>. Citado na página 16.

KHUTSOANE, O.; ISONG, B.; ABU-MAHFOUZ, A. M. Iot devices and applications based on lora/lorawan. In: *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*. [S.l.: s.n.], 2017. p. 6107–6112. Citado na página 17.

KICAD. *KiCad EDA: A Cross Platform and Open Source Electronics Design Automation Suite*. 2019. Disponível em: <https://www.kicad.org/>. Acessado em: 2019-08-13. Citado na página 23.

MCOLLINA. *Mosca: MQTT broker as a module*. 2019. Disponível em: <https://www.mosca.io/>. Acessado em: 2019-08-13. Citado na página 28.

MICROSOFT. *Code editing. Redefined*. 2019. Disponível em: <https://code.visualstudio.com/>. Acessado em: 2019-08-13. Citado na página 28.

MILESIGHT. *Soil Moisture, Temperature and Electrical Conductivity Sensor EM500-SMTC*. 2018. Disponível em: <https://www.milesight-iot.com/lorawan/sensor/em500-smtc>. Accessado em: 2019-05-26. Citado 2 vezes nas páginas 6 e 21.

MILESIGHT. *Discover and Explore Milesight IoT*. 2021. Disponível em: <https://www.milesight-iot.com/>. Acessado em: 2021-07-13. Citado na página 21.

MONGODB. *O banco de dados para aplicativos modernos*. 2019. Disponível em: <https://www.mongodb.com/pt-br>. Acessado em: 2019-08-13. Citado na página 28.

MOZILLA. *JavaScript Tutoriais*. 2019. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>. Acessado em: 2019-08-13. Citado na página 27.

ORG json. *ECMA-404 The JSON Data Interchange Standard.* 2019. Disponível em: <https://www.json.org/json-en.html>. Acessado em: 2019-08-13. Citado na página 28.

PETAJAJARVI, J. et al. On the coverage of lpwans: Range evaluation and channel attenuation model for lora technology. In: . [S.l.: s.n.], 2015. Citado na página 12.

RADIOENGE. *Módulo LoRaMESH.* 2019. Disponível em: <https://www.radioenge.com.br/solucoes/iot/34-modulo-loramesh.html>. Acessado em: 2019-08-13. Citado 2 vezes nas páginas 6 e 25.

Rautmare, S.; Bhalerao, D. M. Mysql and nosql database comparison for iot application. In: *2016 IEEE International Conference on Advances in Computer Applications (ICACA).* [S.l.: s.n.], 2016. p. 235–238. Citado na página 19.

SALESFORCE. *Heroku: Cloud Application Platform.* 2019. Disponível em: <https://www.heroku.com/>. Acessado em: 2019-08-13. Citado na página 27.

SAMACSYS. *Library Loader.* 2019. Disponível em: <https://www.samacsys.com/library-loader/>. Acessado em: 2019-08-13. Citado na página 30.

SENSOTERRA. *SensoTerra Products.* 2019. Disponível em: <https://www.sensoterra.com/en/product>. Acessado em: 2019-08-13. Citado na página 20.

Shi, W. et al. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, v. 3, n. 5, p. 637–646, 2016. Citado na página 16.

Singh, S. Optimize cloud computations using edge computing. In: *2017 International Conference on Big Data, IoT and Data Science (BID).* [S.l.: s.n.], 2017. p. 49–53. Citado na página 16.

SO, J. et al. Loracloud: Lora platform on openstack. In: . [S.l.: s.n.], 2016. p. 431–434. Citado na página 12.

STMICROELECTRONICS. *Ultra-low-power 8-bit MCU with 32 Kbytes Flash, 16 MHz CPU, integrated EEPROM.* 2019. Disponível em: <https://www.st.com/en/microcontrollers-microprocessors/stm8l152c6.html>. Acessado em: 2019-08-13. Citado 2 vezes nas páginas 6 e 24.

SUNDMAEKER, H. et al. Vision and challenges for realizing the internet of things. *Cluster of European Research Projects on the Internet of Things, European Commision*, 04 2010. Citado na página 12.

SYSTEMS, I. *IAR Embedded Workbench for AVR.* 2019. Disponível em: <https://www.iar.com/ewavr>. Acessado em: 2019-08-13. Citado na página 25.

TAURION, C. *Cloud computing-computação em nuvem.* [S.l.]: Brasport, 2009. Citado na página 16.

WEBER, R. H.; WEBER, R. Introduction. In: _____. *Internet of Things: Legal Perspectives.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 1–22. ISBN 978-3-642-11710-7. Disponível em: <https://doi.org/10.1007/978-3-642-11710-7_1>. Citado na página 15.

WORKS, S. *Soil Moisture Monitoring Starter Pack*. 2018. Disponível em: <https://sensorworks.net/precision-agriculture/crop-and-environmental-sensors/vinduino-soil-monitoring-starter-pack>. Accessado em: 2019-05-26. Citado 2 vezes nas páginas 6 e 21.