

**UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS ARARANGUÁ
CENTRO DE CIÊNCIAS, TECNOLOGIAS E SAÚDE
ENGENHARIA DE COMPUTAÇÃO**

HAMILTON LUIZ BEZ BATTI DIAS

**DESENVOLVIMENTO DE UMA PROVA DE CONCEITO BASEADA EM IOTA
PARA RASTREAMENTO DE PRODUTOS EM TRANSPORTE**

Araranguá
2021

Hamilton Luiz Bez Batti Dias

**DESENVOLVIMENTO DE UMA PROVA DE CONCEITO BASEADA EM IOTA
PARA RASTREAMENTO DE PRODUTOS EM TRANSPORTE**

Trabalho de Conclusão de Curso submetido à
Universidade Federal de Santa Catarina para a
obtenção do Grau de Bacharel em Engenharia
de Computação.

Orientador: Prof. Dr. Roderval Marcelino

Araranguá
2021

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Dias, Hamilton
DESENVOLVIMENTO DE UMA PROVA DE CONCEITO BASEADA EM
IOTA PARA RASTREAMENTO DE PRODUTOS EM TRANSPORTE /
Hamilton Dias ; orientador, Roderval Marcelino, 2021.
99 p.

Trabalho de Conclusão de Curso (graduação) -
Universidade Federal de Santa Catarina, Campus Araranguá,
Graduação em Engenharia de Computação, Araranguá, 2021.

Inclui referências.

1. Engenharia de Computação. 2. IOTA. 3. SigFox. 4.
Sistema de Rastreamento. I. Marcelino, Roderval . II.
Universidade Federal de Santa Catarina. Graduação em
Engenharia de Computação. III. Título.

Hamilton Luiz Bez Batti Dias

**DESENVOLVIMENTO DE UMA PROVA DE CONCEITO BASEADA EM IOTA
PARA RASTREAMENTO DE PRODUTOS EM TRANSPORTE**

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de Bacharel e aprovado em sua forma final pelo Curso Engenharia da Computação

Araranguá, 20 de Maio de 2021.

Prof. Fabrício de Oliveira Ourique, Dr.
Coordenador do Curso

Banca Examinadora:

Prof. Roderval Marcelino, Dr.
Orientador
Universidade Federal de Santa Catarina

Prof. Martin Augusto Gagliotti Vigil, Dr.
Avaliador
Universidade Federal de Santa Catarina

Prof. Jim Lau, Dr.
Avaliador
Universidade Federal de Santa Catarina

AGRADECIMENTOS

Agradeço principalmente à minha família, especialmente a minha mãe Bianca e ao meu pai Hamilton. Agradeço também a todos os meus amigos, professores e a todas as outras pessoas que me ajudaram. Agradeço ao meu orientador Roderval e ao Wellington que me deram a oportunidade de fazer parte deste projeto.

RESUMO

Com o mundo cada vez mais globalizado, o sucesso de muitas empresas depende do seu produto competir no maior número possível de localizações do mundo; com isso, a distância entre produtor e consumidor fica cada vez mais maior. Logo, manter a segurança, a qualidade, o histórico e a localização do produto no caminho inteiro se tornou um grande desafio. As soluções utilizadas por algumas empresas na atualidade não conseguem atingir os três principais pontos de um bom sistema de rastreabilidade, que são: segurança, confiabilidade e transparência. Já existem soluções que atingem dois dos três pontos, por isso técnicas e pesquisas que visam atingir os três pontos em um sistema coeso têm sido sugeridas e produzidas com base em novas tendências tecnológicas. Neste trabalho, é proposto um modelo de sistema de rastreamento baseado em um dispositivo embarcado em um produto em transporte que utiliza a rede SigFox como meio de comunicação e uma aplicação Web que utiliza o *Tangle* da IOTA como banco de dados e serve como plataforma de visualização dos dados obtidos. A partir do modelo proposto foi escolhido um protótipo de dispositivo de monitoramento e desenvolvido um protótipo da aplicação, no qual foram realizados testes para simular uma aplicação real, fazendo com que o dispositivo fosse enviado a várias cidades para consultar os dados obtidos. Foi possível obter ambos os dados dentro das cidades, sendo possível ver a trajetória e o gráfico de temperatura do percurso, tendo apenas poucos resultados de localização em áreas não urbanas.

Palavras-chave: Sistema de rastreamento. Rede SigFox. *Tangle*. aplicação Web.

ABSTRACT

With the world increasingly globalized, the success of lots of companies around the world depends on its products being able to compete in as many locations as possible, with that, the distance between producers and consumers is growing each day. Then, keeping security, quality, history and localization of the product in the route has become a great challenge. The utilized solutions of some companies nowadays can not achieve the three main aspects of a good tracking system, that are: security, confiability and transparency. There are some solutions that can achieve two of them, because of that, techniques and research are being made to achieve all three aspects in a cohesive system that have been suggested and developed based on new technological trends. In this work is proposed a new tracking system model based on an embedded system on a moving product, that uses SigFox's network to communicate and a web application that uses IOTA's Tangle as database and is used as a platform to see obtained data. Starting by the proposed model, it was chosen as a device and developed a web application prototype, where tests were done to simulate a real application, making the device travel through cities to see the obtained data. It was possible to obtain both the variables inside the cities, making it possible to see the trajectory and temperature graph route, but there were few position results on non urban areas.

Keywords: Tracking system. SigFox network. Tangle. Web application.

LISTA DE FIGURAS

Figura 1 - Fluxograma de desenvolvimento	18
Figura 2 - Código de barras no padrão EAN-13	21
Figura 3 - Pacotes de iogurte com o QR Code à vista	23
Figura 4 - História da tecnologia de comunicação de celulares	25
Figura 5 - Comparação de características	29
Figura 6 - Arquitetura SigFox e LoRa	30
Figura 7 - Formato da mensagem LoRaWAN	32
Figura 8 - Formato do pacote do Sigfox	33
Figura 9 - Arquitetura da rede SigFox detalhada	35
Figura 10 - Modem da SigFox	36
Figura 11 - Pinagem do modem	37
Figura 12 - Conexão modem e microcontrolador	37
Figura 13 - Os três casos comuns de chamadas	40
Figura 14 - Arquitetura Baseada em Objetos	41
Figura 15 - Arquitetura publisher/subscriber	42
Figura 16 - Exemplos de entrada e saída de uma função hash	43
Figura 17 - Estrutura do Blockchain	43
Figura 18 - Soft Fork	45
Figura 19 - Exemplos de projetos reais utilizando blockchain	46
Figura 20 - Infográfico sobre a decisão de ter ou não uma blockchain	48
Figura 21 - Exemplo de um DAG	49
Figura 22 - DAG com pesos das transações	50
Figura 23 - Exemplo de documento	54
Figura 24 - Downloads de ferramentas para o front-end	55
Figura 25 - Exemplo da Virtual DOM	56
Figura 26 - Arquitetura da solução	58
Figura 27 - Arquitetura da aquisição de sinais	60
Figura 28 - Dispositivo WS2	61
Figura 29 - Zonas SigFox	62
Figura 30 - Arquitetura da Nuvem	63
Figura 31 - Arquitetura do Software	65
Figura 32 - Exemplo de JSON enviado ao Tangle	66
Figura 33 - Inserção no Tangle	67
Figura 34 - Chamada da função para inserir no Tangle	67
Figura 35 - Função para inserir no Tangle	68
Figura 36 - Coleções do banco de dados	69
Figura 37 - Tela inicial do sistema	70

Figura 38 - Dashboard.	71
Figura 39 - Cadastro de viagens.	71
Figura 40 - Leitura do dado do Tangle.	72
Figura 41 - Temperatura da entrega.	72
Figura 42 - Localização da entrega.	73
Figura 43 - Dados de temperatura.	74
Figura 44 - Dados de localização.	74
Figura 45 - Gráfico de temperatura do segundo teste.	76
Figura 46 - Localização do segundo teste.	76

LISTA DE QUADROS

Quadro 1 - Características dos quatro tipos de código de barras 2D mais populares	21
Quadro 2 - Comparação dos três primeiros sistemas	23
Quadro 3 - Comparação de preços	28
Quadro 4 - Zonas da SigFox	34
Quadro 5 - Comparação entre PHP e Node.....	56

LISTA DE ABREVIATURA E SIGLAS

ACK	<i>Acknowledgement</i>
ADC	<i>Analogic Digital Converter</i>
AIDS	<i>Acquired Immunodeficiency Syndrome</i>
API	<i>Application Programmable Interface</i>
BDMA	<i>Beam Division Multiple Access</i>
BFT	<i>Byzantine Fault Tolerance</i>
BPSK	<i>Binary Phase-Shift Keying</i>
CDMA	<i>Code Division Multiple Access</i>
CEPT	<i>Conference of European Posts and Telegraphs</i>
CSS	<i>Chirp Spread Spectrum</i>
DAG	<i>Directed Acyclic Graph</i>
DOM	<i>Document Object Model</i>
EAN	<i>European Article Number</i>
ECC	<i>Error Correcting Code</i>
EEPROM	<i>Electrically Erasable Programmable Read-Only Memory</i>
eMBB	<i>Enhanced Mobile Broadband</i>
FDMA	<i>Frequency Division Multiple Access</i>
GSM	<i>Global System for Mobile Communication</i>
HTTP	<i>HyperText Transfer Protocol</i>
I/O	<i>In/Out</i>
ID	<i>Identifier</i>
IETF	<i>Internet Engineering Task Force</i>
IoT	<i>Internet of Things</i>
IP	<i>Internet Protocol</i>
ISM	<i>Industrial Scientific and Medical</i>
ITU	<i>International Telecommunication Union</i>
JSON	<i>Javascript Object Notation</i>
LAN	<i>Local Area Networks</i>
LTE	<i>Long Term Evolution</i>
LoRa	<i>Long Range</i>
LPWAN	<i>Low Power Wide Area Networks</i>
MAC	<i>Media Access Control</i>
MAM	<i>Masked Authenticated Messaging</i>
MCMC	<i>Markov Chain Monte Carlo</i>
MERN	<i>MongoDB, Express, React, Node</i>
MIMO	<i>Massive Multiple-input multiple-output</i>
mMTC	<i>Massive Machine Type Communications</i>
MTU	<i>Maximum Transmission Unit</i>
NB-IoT	<i>Narrow Band Internet of Things</i>
NPM	<i>Node Package Manager</i>

OFDMA	<i>Orthogonal Frequency Division Multiple Access</i>
P2P	<i>peer-to-peer</i>
PoW	<i>Proof of Work</i>
QoS	<i>Quality of Service</i>
QPSK	<i>Quadrature Phase-Shift Keying</i>
QR	<i>Quick Response</i>
RFID	<i>Radio Frequency Identification</i>
RSSI	<i>Received Signal Strength Indicator</i>
SF	<i>spreading factors</i>
SQL	<i>Structured Query Language</i>
SSID	<i>Service Set Identifier</i>
TCP	<i>Transmission Control Protocol</i>
TDMA	<i>Time Division Multiple Access</i>
TSA	<i>Tip Selection Algorithm</i>
UART	<i>Universal Asynchronous Receiver Transmitter</i>
UNB	<i>Ultra Narrow Band</i>
URL	<i>Uniform Resource Locator</i>
URLLC	<i>Ultra Reliable and Low Latency Communication</i>
WI-FI	<i>Wireless Fidelity</i>
WAN	<i>Wide Area Networks</i>

SUMÁRIO

INTRODUÇÃO	16
1.1 Objetivos	17
1.1.1 Objetivo Geral	17
1.1.2 Objetivos Específicos	17
1.2 Metodologia	18
1.3 Organização do Trabalho	18
RASTREABILIDADE DE PRODUTOS	20
2.1 Sistemas de Rastreabilidade de Produtos	20
2.1.1 Código de Barras	21
2.1.2 QR Code	22
2.1.3 RFID	23
2.1.4 Redes de Celulares	25
3. REDES LPWAN	28
Quadro 3 - Comparação de preços	29
3.1 NB-IoT	30
3.2 LoRaWAN	31
3.3 SigFox	32
3.4 Sistema Embarcado para LPWAN	36
4. SISTEMAS DISTRIBUÍDOS	38
4.1 Arquiteturas de Sistemas Distribuídos	39
4.1.1 Arquitetura por Camadas	39
4.1.2 Arquitetura Baseada em Recursos	40
4.1.4 Arquitetura Baseada em Eventos	41
4.2 Aplicações	42
4.2.1 Blockchain	42
4.2.1.1 Casos que o Blockchain não é Necessário	47
4.2.2 Tangle-IOTA	49
5. SISTEMAS WEB	53
5.1 MongoDB	53
5.2 Express	54
5.3 React	55
5.4 Node	56
6. METODOLOGIA DESENVOLVIDA	57
6.1.1 Requisitos Funcionais	59
6.1.2 Requisitos não Funcionais	59

6.2 Aquisição de Sinais	59
7. TESTES E VALIDAÇÕES	74
8. CONSIDERAÇÕES FINAIS	78
8.1 Trabalhos Futuros	78
REFERÊNCIAS	80
APÊNDICES	86
APÊNDICE A - TABELA DE LOCALIZAÇÃO DO PRIMEIRO TESTE	86
APÊNDICE B - TABELA DE TEMPERATURA DO PRIMEIRO TESTE	91
APÊNDICE C - TABELA DE LOCALIZAÇÃO DO SEGUNDO TESTE	98
APÊNDICE B - TABELA DE TEMPERATURA DO SEGUNDO TESTE	99

1. INTRODUÇÃO

Com a globalização, o sucesso de muitas empresas depende do seu produto conseguir competir em todas as partes do mundo; com isso, a distância entre produtor e consumidor pode ser enorme. Logo, manter a segurança, a qualidade, o histórico, a localização do produto no caminho inteiro se tornou um grande desafio. Nos últimos anos, devido a esse desafio, a indústria de rastreabilidade perdeu muito a sua credibilidade por causa de atrasos, perdas de mercadorias, perda de qualidade durante o trajeto, falsificação de informações por parte dos atores e outros motivos (AUNG; CHANG, 2014; MISHRA *et al.*, 2018).

A integridade dos produtos não significa apenas a sua segurança, mas também a origem e todo o percurso, pois também há possibilidade de fraude nos dois últimos. Para buscar garantir a segurança, é preciso de um sistema de rastreabilidade que possibilite que todos os atores (nesse caso, seriam os produtores, consumidores, distribuidor, fornecedor e governo) possam verificar todas as variáveis de qualidade e integridade do produto (AUNG; CHANG, 2014; DABBENE; GAY; TORTIA, 2014).

Em entrevista para a Revista Forbes, David Sonstebo disse que 4 trilhões de dólares são perdidos por ano em produtos perdidos e falsificados. Isso sem falar nas horas perdidas procurando, catalogando e verificando documentos (FENECH, 2018).

Uma das tentativas de soluções foi o RFID (*Radio Frequency Identification*, ou em português, Identificação de Frequência de Rádio) e um sistema *web*, no qual as informações durante todo o processo seriam armazenadas em um banco de dados. Esse processo teve dois problemas: o primeiro é a leitura do RFID, que obteve um sucesso de 80% no momento de obter as grandezas físicas, e o segundo é a segurança de um banco de dados comum, que não seria a ideal para algo desse porte (ALFIAN *et al.*, 2017).

Em 1991, foi a primeira vez que uma estrutura de blocos criptografados e conectados entre si foi idealizada, mas, só em 2008, com o surgimento e sucesso do *Bitcoin* que essa estrutura se popularizou. Essa estrutura é conhecida como *blockchain*. Essa tecnologia consiste em blocos com informações, em que cada bloco consegue verificar a informação do próximo bloco, fazendo assim uma corrente que impede a falsificação de informação. *Blockchains* são seguras, transparentes e, o mais importante, são descentralizadas, ou seja, não é necessário confiar em uma entidade central para garantir a validade da informação. Desde 2015, algumas soluções para a rastreabilidade de produtos estão sendo desenvolvidas utilizando *blockchains* (GÁLVEZ; MEJUTO; SIMAL-GÁNDARA, 2018).

Mas essa tecnologia apresenta algumas limitações. A primeira delas é a velocidade, o *Bitcoin* é um bom exemplo, no qual só é possível realizar sete transações por segundo. O segundo é o preço, uma boa parte dos *blockchains* são caros (tanto em energia, quanto em processamento, quanto em tempo) para adicionar uma nova transação à rede. A tecnologia já recebeu vários tipos de ataques, sendo o “ataque dos 50%” o mais famoso. Por esses três motivos, a tecnologia se torna inviável para ser utilizada mundialmente na rastreabilidade de produtos (PECK, 2017).

Em 2015, foi publicado um *paper* chamado “*The Tangle*”, que explica um novo tipo de *blockchain*, no qual se abandona a ideia de um bloco ligado com outro e traz a ideia de um grafo direcionado acíclico, ou seja, cada bloco consegue verificar a informação de outros dois

blocos. Fazendo assim com que a segurança continue, mas principalmente aumente muito a velocidade com que novas informações possam ser inseridas (DIVYA; BIRADAR, 2018).

A principal tecnologia que utiliza esse *Tangle* é a IOTA, uma criptomoeda feita para *internet* das coisas que, além de melhorar a velocidade, também consegue melhorar mais coisas do *Bitcoin*: A IOTA é leve, então processadores mais fracos e lentos conseguem inserir informações. A IOTA também não tem custo para inserir novas informações. Isso faz com que o consumo de energia de dispositivos que inserem informações seja muito baixo, o que a torna uma opção mais atrativa para o problema proposto por esse trabalho (DIVYA; BIRADAR, 2018).

Para fazer a comunicação desses dispositivos de baixo consumo energético, é necessária uma forma de comunicação que suporte tais condições. Para isso serve as LPWANs (*Low Power Wide Area Networks*, ou em português, redes de área ampla e baixa potência). Na qual se pode destacar três principais: SigFox, LoRA e NB-IoT (MEKKI *et al.*, 2019).

Baseado no que foi exposto nesta seção, este trabalho propõe um método de rastreamento de produtos utilizando os conceitos de IoT por meio de dispositivos conectados por uma infraestrutura LPWAN em conjunto com o *Tangle* para guardar e manter seguras as informações. Dentro do modelo proposto, foi desenvolvido um protótipo utilizando a rede SigFox, o *Tangle* da IOTA e uma aplicação *web* como conexão e ambiente de visualização e análise de dados.

1.1 Objetivos

Esta subseção apresenta o objetivo geral e os objetivos específicos desta monografia de conclusão de curso.

1.1.1 Objetivo Geral

Desenvolver um protótipo de aplicação Web como prova de conceito baseada no *Tangle* da IOTA usando SigFox para comunicação e rastreamento de produtos em transporte.

1.1.2 Objetivos Específicos

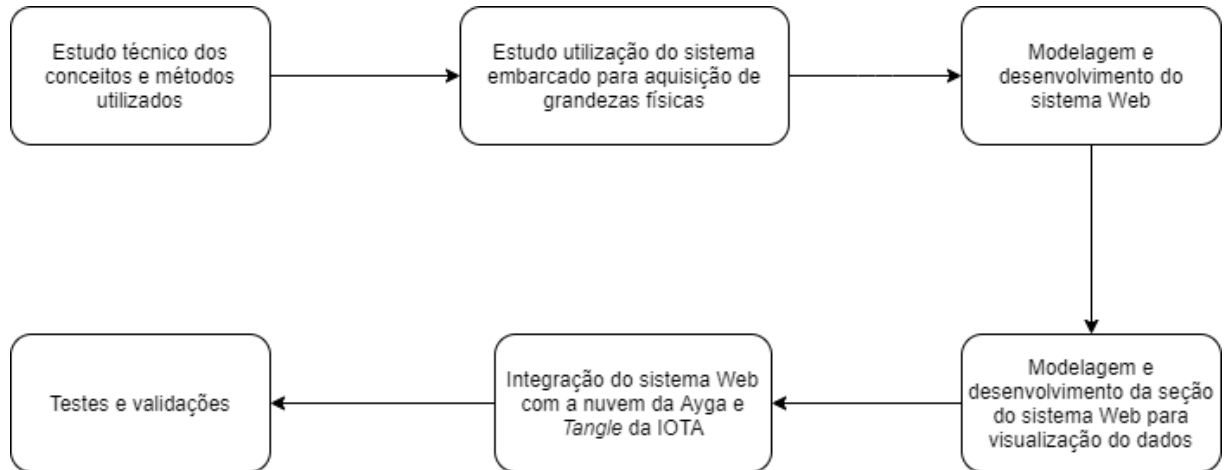
Considerando o desenvolvimento do trabalho e o objetivo geral apresentado, destacam-se os seguintes objetivos específicos:

- Utilizar um sistema embarcado para leitura e transmissão de grandezas físicas;
- Desenvolver um sistema *web* para recebimento, envio e visualização de dados;
- Integrar a *SigFox Cloud* ao sistema *web* para recebimento de dados;
- Integrar o *Tangle* da IOTA ao sistema *web* para armazenamento e segurança dos dados.

1.2 Metodologia

A metodologia empregada para o desenvolvimento do trabalho é segmentada em seis etapas, que são apresentadas na Figura 1.

Figura 1 - Fluxograma de desenvolvimento



Fonte: Elaborado pelo Autor.

A primeira parte consiste na pesquisa bibliográfica referente aos conceitos e às tecnologias abordadas no presente trabalho. No qual se salientam os sistemas de rastreabilidade atuais de produtos, sistemas de aquisição de grandezas físicas, redes LPWAN, *blockchain*, o *Tangle* da IOTA e tecnologias *web*.

Na segunda etapa do trabalho, foi necessário estudar utilizar um sistema embarcado que possibilitasse obter dados de uma determinada carga, como a localização, temperatura, umidade e outras grandezas mais detalhadas nas seções seguintes do trabalho.

Na terceira parte do trabalho, foi necessário projetar um sistema *web* que recebesse dados da *Cloud* da *SigFox* e envie-os para o *Tangle* da IOTA.

A quarta parte foi constituída para desenvolver a aplicação *web* e integrar as três partes: Aplicação *web*, *Cloud* da *SigFox* e *Tangle* da IOTA.

Ainda foi necessário projetar um sistema *web* para capturar e mostrar os dados obtidos na *Tangle* da IOTA. Isso foi realizado na quinta parte do trabalho.

A penúltima parte do trabalho foi desenvolver o sistema *web* do passo anterior e integrar com o *Tangle*.

Por fim, foram realizados os testes de validação de todo o protótipo, o qual avaliou a correta obtenção das grandezas físicas, o correto recebimento dos dados e a correta visualização dos dados.

1.3 Organização do Trabalho

O presente trabalho está organizado da seguinte forma:

O **Capítulo 1** apresenta o contexto da problemática, assim como os objetivos geral e específicos junto com os procedimentos metodológicos adotados para o desenvolvimento do trabalho.

O **Capítulo 2** cita algumas soluções existentes para o problema da rastreabilidade de produtos, junto com os seus defeitos e pontos positivos de cada solução. Cada solução apresentada é exemplificada com modelos reais e testados.

O **Capítulo 3** aborda sobre as redes LPWAN, apresentando os seus três principais exemplos e suas características. Também foi realizado estudo sobre a LoRaWAN, SigFox e NB-IoT, que são as três principais tecnologias de redes LPWAN. Este capítulo também apresenta exemplos de *hardware* utilizados para se comunicar com redes LPWAN.

O **Capítulo 4** descreve e define o que é um sistema distribuído. Além disso, também comenta e estuda as principais arquiteturas. Duas tecnologias são apresentadas, a *Blockchain* e o *Tangle*, ilustrando as vantagens e desvantagens de utilizar alguma delas.

O **Capítulo 5** trata sobre as principais tecnologias Web para o desenvolvimento da prova de conceito deste trabalho, no qual é abordado sobre MongoDB, Express, React e NodeJS.

O **Capítulo 6** tem como objetivo descrever o modelo proposto, apresentando os principais componentes do sistema (arquitetura da solução) e como eles se comportam e comunicam-se.

O **Capítulo 7** ilustra como a solução proposta foi testada e validada. Também são apresentados os resultados de tais testes e validações.

O **Capítulo 8** apresenta as considerações finais a partir do que foi desenvolvido neste trabalho.

2. RASTREABILIDADE DE PRODUTOS

Neste capítulo, são apresentados os conceitos básicos de sistemas de rastreabilidade de produtos, trazendo, também, as tecnologias que já foram utilizadas nos últimos anos, fazendo comparações e análises entre elas.

2.1 Sistemas de Rastreabilidade de Produtos

As definições de “rastreabilidade” e “sistema de rastreabilidade” podem ser muito abrangentes na literatura. Mas, em todos os casos, os autores se tangem à habilidade de garantir o monitoramento e o registro de produtos desde a sua fabricação até a sua entrega. Monitoramento é a característica de seguir o produto durante todo o seu percurso. Registro é o poder de determinar a origem e as características do produto (DABBENE; GAY; TORTIA, 2014). São nessas definições que esse projeto se baseia e é desenvolvido.

A habilidade de rastrear todo o percurso, coletar dados rigorosos e também obter qualquer erro de alguma variável durante o transporte de uma determinada mercadoria é de muita importância para empresas e governos na atualidade. Essa complexidade e importância se deve principalmente a regulações, certificações e padrões, tanto locais quanto internacionais. Quando na falta de obtenção de um requerimento pode colocar em risco a qualidade de nomes e marcas e, pior ainda, causar algum tipo de dano à pessoa do cliente, como doenças em casos de transporte de comida (DABBENE; GAY; TORTIA, 2014).

Uma maneira de rastrear produtos de forma eficiente só pode ser criada e alcançada com uma combinação de fatores. Em que cada ator da cadeia precisa conhecer o outro, por exemplo, o fornecedor precisa conhecer o vendedor e vice-versa para gerar confiança. Para conseguir alcançar um bom nível de confiança nas redes de comércio atuais, que são complexas, faz-se necessário o uso de tecnologias (THAKUR *et al.*, 2009).

Para tanto o sistema de rastreabilidade necessita estar sustentado em quatro pilares: identificação do produto, dados a serem monitorados, percurso do produto e ferramentas utilizadas. Esses quatro pontos precisam estar bem estruturados antes de começar a pensar na visualização e modelagem dos dados obtidos (THAKUR *et al.*, 2009).

Um dos grandes desafios de um sistema de rastreabilidade é a troca de informações entre atores de uma forma padronizada e de fácil entendimento, sem comprometer a privacidade de um dos atores. Outras duas complicações que precisam ser levadas em consideração são: primeiro cada produto tem as suas peculiaridades e isso torna difícil a obtenção de um modelo geral; segundo a disponibilidade de dados de todos os atores, já que a qualidade do sistema depende de um trabalho conjunto de todos (HU J. *et al.*, 2013).

A automatização de coleta de dados melhora a precisão e a confiabilidade no monitoramento de um produto. Por isso, cada vez mais empresas e governos estão investindo em novas tecnologias para suprir esses problemas. Pode-se utilizar como exemplo o código de barras, *QR Code*, *Radio Frequency Identification* (RFID), em que todas essas tecnologias são tentativas de melhoramento na área (DABBENE; GAY; TORTIA, 2014).

2.1.1 Código de Barras

Código de barra é uma representação gráfica em uma dimensão de dados numéricos, na qual a leitura das informações precisa de um tipo específico de *scanner*, que converte as barras em números que humanos conseguem compreender (TARHAN *et al.*, 2014).

Os códigos de barras têm muitas padronizações, mas a mais utilizada mundialmente é a padronização européia, chamada de EAN-13 (*European Article Number*). Uma padronização de código de barras define os detalhes técnicos da identificação, por exemplo a largura das barras, método de codificação e caracteres (YOUSUF *et al.*, 2019).

Figura 2 - Código de barras no padrão EAN-13



Fonte: Adaptado de Yousuf *et al.* (2019).

A Figura 2 apresenta um código de barras no padrão EAN-13, no qual os dois primeiros dígitos representam a nacionalidade do fabricante. Os próximos cinco dígitos representam o código do fabricante que a autoridade responsável atribuiu ao fabricante. Os próximos cinco dígitos são o ID que o fabricante escolheu para o produto. O último número é utilizado pelo *scanner* para fazer a checagem de todos os outros (YOUSUF *et al.*, 2019).

Em um sistema de rastreabilidade baseado em códigos de barras, é necessária a interferência de seres humanos em várias etapas do processo. A primeira interferência é na hora de ler o código de barras. Pois o *scanner* tem que ficar em uma posição exata para conseguir ler. A segunda é que as informações são inseridas manualmente no sistema que gera o código de barra, fazendo com que dados sejam facilmente alterados (REGATTIERI; GAMBERI; MANZINI, 2007).

O principal benefício do código de barras unidimensional é o preço. Essa tecnologia possui o preço mais baixo para ser implementada em vários mercados. Mas essa também tem a capacidade de armazenamento muito baixa, sendo apenas números que podem ser armazenados (REGATTIERI; GAMBERI; MANZINI, 2007).



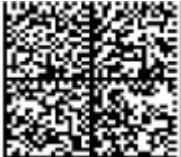
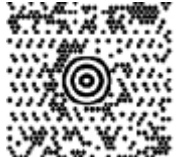
Um exemplo foi estudado por Yousuf *et al.* (2019), no qual os códigos de barras foram inseridos nas etiquetas que ficam nas orelhas de bois e vacas. Apesar de ter facilitado um pouco o controle do gado, foram encontradas algumas dificuldades. As duas principais foram o trabalho manual que continuou forte na fazenda, pois os fazendeiros ainda precisavam ir até cada animal para ler o código de barras, mas também muitos desses códigos ficaram impossíveis de serem lidos por causa da sujeira do ambiente aberto.

2.1.2 QR Code

O *Quick Response (QR) Code* é muito parecido com o código de barras, porém pode armazenar mais informações que seu antecessor. Esse sistema muitas vezes é chamado de “códigos de barras 2D”, pelo fato que, enquanto o código de barras só consegue guardar informações na horizontal, o *QR Code* consegue armazenar informações tanto na horizontal quanto na vertical (TARHAN *et al.*, 2014).

Existem mais de vinte tipos de códigos de barras 2D. No Quadro 1, pode-se observar características dos quatro mais populares, sendo o *QR Code* o mais utilizado de todos, pois além de ser graça para todo mundo criar o seu, ele é o código de barras 2D que consegue armazenar a maior quantidade de informações (QUERINI *et al.*, 2011).

Quadro 1 - Características dos quatro tipos de código de barras 2D mais populares

Tecnologia	QR Code	PDF417	DataMatrix	MaxiCode
Imagem				
Desenvolvedor (País)	DENSO (Japão)	Symbol Technologies (EUA)	RVSI Acuity CiMatrix (EUA)	UPS (EUA)
Tipo	Matriz	Código de barras empilhado	Matriz	Matriz
Dados Numéricos (dígitos)	7.089	2.710	3.116	138
Dados Alfanuméricos (caracteres)	4.296	1.850	2.355	93
Dados Binários (dígitos)	2.953	1.018	1.556	-
Principais Características	Muita capacidade, ocupa pouco espaço impresso, alta velocidade para leitura	Muita capacidade	Ocupa pouco espaço impresso	Alta velocidade para leitura

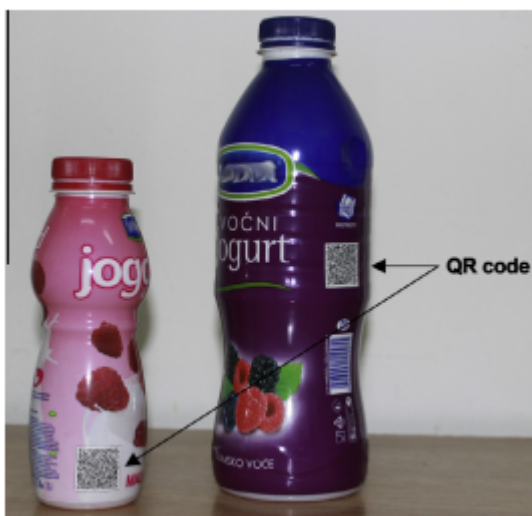
Fonte: Adaptado de Querini (2011).

O *QR Code* também ocupa menos espaço que o código de barras e, mesmo com menos espaço, consegue armazenar muito mais informações. Esse código de barras em duas dimensões não tem direção correta para ser lido, o que diminui a interferência humana e os erros de leitura. Outra vantagem é a praticidade de leitura, porque o sistema não precisa mais

de um *scanner* especial, ou seja: com um celular comum é possível obter as informações (TARHAN *et al.*, 2014).

Tarhan *et al.* (2014) apresentam um exemplo de como o sistema baseado em *QR Code* funciona, escolhendo como produto o iogurte de frutas. Nesse estudo, os autores colocam um *QR Code* único em cada unidade, e todas as informações de todos os produtos são guardadas em um banco de dados central. Mas com o ID que está no *QR Code* é possível acessar as informações do produto por meio da internet. A Figura 3 mostra exemplos desse QRCode.

Figura 3 - Pacotes de iogurte com o *QR Code* à vista



Fonte: Tarhan (2014, n.p.).

Antes de ter o produto pronto à venda, é necessário obter as suas informações para inserir no banco de dados e posteriormente no *QR Code*. O primeiro ponto de coleta de informações é nos tanques de leite dos fazendeiros, onde a qualidade do leite é determinada. Nesse ponto, é possível obter a localização da fazenda, a data de armazenamento, a quantidade armazenada e outras variáveis que influenciam na qualidade final do produto. O segundo ponto é na coleta da fruta do iogurte, quando é possível obter as mesmas variáveis de controle. Com esses dois pontos é possível gerar os dados para a fábrica gerar os *QR Codes* de cada lote e, assim, ajudar o consumidor a escolher melhor o seu iogurte (TARHAN *et al.*, 2014).

O *QR Code* consegue ainda funcionar com um pouco de sujeira. Se cerca de 30% do código estiver coberto de sujeira, ainda será possível obter as informações codificadas dentro dele (QUERINI *et al.*, 2011). Mas, apesar das novas vantagens, a matriz de dados ainda não consegue armazenar novos dados depois de criada e por causa disso fica inviável obter informações relacionadas ao transporte do produto (FAN *et al.*, 2019).

2.1.3 RFID

Radio Frequency Identification (RFID) usa campos eletromagnéticos de radiofrequência para transferir dados usando *tags* que estão presas nas mercadorias. Algumas

dessas *tags* são chamadas de *tags* passiva, na qual recebem energia e podem ser lidas por uma distância pequena por meio de indução eletromagnética. O segundo tipo de *tag*, chamado de *tag* ativo, tem fonte própria de energia (normalmente uma bateria) e podem ser lidas as informações por meio através de dezenas de metros de distância. Diferente do código de barras, o RFID não precisa de linha de visão ou contato físico para ser lido (SEGURA, 2016).

O RFID dá um grande salto também em relação à memória. Enquanto que os antecessores apenas guardavam algum tipo de identificador, o RFID vem com memória utilizável, como a EEPROM (*Electrically Erasable Programmable Read-Only Memory*) que varia entre 256 bits e 64 kBytes. Com esse adicional de memória, nem todas as variáveis precisam ser inseridas em um servidor central, possibilitando uma descentralização da informação dos produtos (SEGURA, 2016).

Uma comparação mais simples das três tecnologias apresentadas até agora neste capítulo é apresentada no Quadro 2.

Quadro 2 - Comparação dos três primeiros sistemas

Tecnologia	Código de Barras	Código de Barras 2D	RFID
Condições Ambientais	Sensível ao ambiente, como sujeira e arranhões	Sensível ao ambiente, como sujeira e arranhões	Durável, à prova d'água.
Segurança	Baixa (sem criptografia)	Alta (criptografia simples)	Altíssima (alta criptografia)
Distância de Leitura	Perto	Perto	Longe (100m)
Capacidade de Armazenamento	Apenas números	Números, alfabeto e alguns caracteres especiais	Qualquer tipo de caractere
Leitura/Escrita	Não é atualizável	Não é atualizável	Novas informações podem ser inseridas
Preço	Barato	Barato	Caro

Fonte: Fan (2019, n.p.).

No estudo de Metzner, Fray e Cugnasca (2015), foi testado um modelo de sistema de rastreamento de medicamentos de alto valor (medicamentos para câncer, AIDS e outras doenças que precisam de medicação contínua) utilizando RFID. Nesse modelo, é inserida uma *tag* de RFID na embalagem do medicamento, o qual é rastreado desde a unidade produtiva até o cliente final.

Na fabricação, a *tag* recebe as seguintes variáveis: número de identificação, fornecedor da matéria prima, data em que o remédio foi fabricado, lote que pertence e pessoas que foram responsáveis por esse processo. O segundo ponto de leitura e escrita na *tag* é no centro de distribuição, onde todos os medicamentos requisitados pelo mesmo cliente são unificados em uma única *tag*, contendo todas as informações anteriores. Também são inseridas a hora que o lote saiu do centro, o caminhão, trajeto e destino (METZNER; FRAY; CUGNASCA, 2015).

Chegando, por fim, à gôndola na qua esses medicamentos são armazenados e controlados e podem ser retirados pelo cliente final, que pode ser um médico, enfermeiro ou paciente, com a devida autorização. Dessa forma, o cliente final tem acesso a todos os processos que esse medicamento passou, tendo assim, a certeza da qualidade do produto (METZNER; FRAY; CUGNASCA, 2015).

Como dificuldade no modelo apresentado, pode-se considerar que:

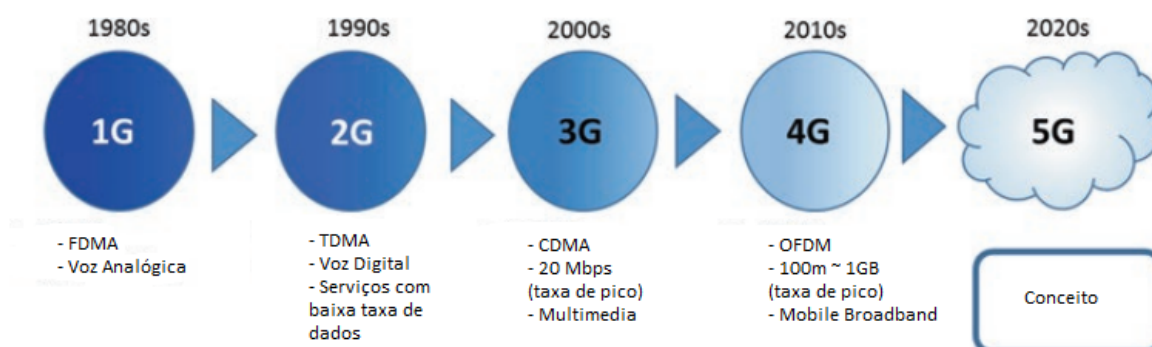
As principais limitações dizem respeito a: dificuldade em coletar dados de operações, a complexidade do tema, falta de padronização de uma metodologia para elaboração de sistemas de rastreabilidade e a falta de padronização das formas de transmissão de dados das tecnologias utilizadas. (METZNER; FRAY; CUGNASCA, 2015, p. 9).

Outro problema foi observado quando se usa RFID em produtos ou embalagens de metais. Como o RFID utiliza um campo eletromagnético para fazer a troca de energia e informações, quando esse campo se encontra com uma superfície de metal, pode diminuir drasticamente a distância de comunicação e, também, tem uma possibilidade de alterar os dados enviados (SEGURA, 2016).

2.1.4 Redes de Celulares

As redes de celulares começaram a ser desenvolvidas no início dos anos 1970. A primeira geração tinha como método de acesso de canal a *Frequency Division Multiple Access* (FDMA) e também um serviço de voz analógica para os usuários. Dez anos depois, chegou a 2G, o método de acesso de canal mudou para o *Time Division Multiple Access* (TDMA) que tinha como principal serviço a voz digital, como também outros serviços de baixa taxa de dados. O *Code Division Multiple Access* (CDMA), desenvolvido para a terceira geração em meados de 2000, possibilitou que mais usuários utilizassem a mesma rede, pois agora a banda de rede não era mais dividida igualmente o tempo todo, e sim dividida apenas quando utilizada. A Figura 4 ilustra a evolução das redes de celulares (LEI, 2020).

Figura 4 - História da tecnologia de comunicação de celulares



Fonte: Lei (2020, n.p.).

Em 2005, foi iniciado o desenvolvimento da quarta geração (4G). Foi desenvolvido a *Orthogonal Frequency Division Multiple Access* (OFDMA, ou só OFDM), que utiliza a banda dividida em múltiplas portadoras ortogonais (subportadoras), para que as bandas não interfiram umas nas outras. Fazendo com que a rede tenha a mesma taxa de transferência, mas muito mais confiável que as antecessoras. Pode, dessa maneira, fazer com que muitos usuários pudessem conectar-se e ainda manter uma velocidade de até 1 GB em condições ideais, que é chamado de *Mobile Broadband* (banda larga móvel, em português) (LEI, 2020).

Um sistema de rastreabilidade foi testado por Chen (2020) no mercado de porcos da China. Nele foram escolhidas as redes de celulares, pois o país é muito grande e ficaria muito caro implementar outra tecnologia, já que essas redes já cobrem uma boa parte do território. Com o auxílio de *QR Codes* e aplicativos já existentes, os atores do mercado (como fazendeiros, abatedouro, transportadora e mercado) podem inserir novas informações em um banco de dados centralizado. Ou seja, a cada etapa que o produto percorre, o ator escaneia com seu próprio celular o *QR Code*, confirma as informações sobre o produto e insere as suas informações em específico. Isso foi implementado com sucesso na China, pois, independentemente da localização dos atores, a rede de celular estava disponível para inserção e leitura de informações no banco de dados central. Como resultado, o consumidor obteve as informações de seu produto, quase em tempo real.

No momento que esse trabalho está sendo escrito, o 5G ainda não foi lançado. Mas o seu desenvolvimento segue uma ideia principal, que é conectar o mundo inteiro com comunicação onipresente e contínua. Para que essa ideia seja alcançada, a *International Telecommunication Union* (ITU) estipulou três pilares para as tecnologias que podem fazer parte do desenvolvimento do 5G se basearem: *Enhanced Mobile Broadband* (eMBB) para aumentar a capacidade das redes, *Massive Machine Type Communications* (mMTC) para conseguir se comunicar com inúmeros dispositivos de baixa capacidade e *Ultra Reliable and Low Latency Communication* (URLLC) para melhorar os sistemas em tempo real (PARIKH; BASU, 2020).

Uma tecnologia que é uma forte candidata a estar embutida no 5G é a *Massive Multiple-input multiple-output* (MIMO), que possibilita o envio e recebimento de dados de várias fontes em um mesmo canal de rádio. Enquanto a MIMO do 4G utilizava uma configuração com até quatro antenas, a *massive MIMO* utiliza 64 para recebimento de dados e 64 para envio de dados, possibilitando assim muitos mais usuários em um mesmo canal (PARIKH; BASU, 2020).

Para que o *massive MIMO* fosse criado, foi necessário o desenvolvimento de um novo algoritmo de acesso ao canal para envio de dados. O *Beam Division Multiple Access* (BDMA) supriu essa necessidade. Com esse algoritmo, o sistema consegue detectar a localização de onde estão vindo mais requisições na área que as antenas cobrem. Assim, as antenas conseguem alocar parte dos seus sinais para cobrir essa área com maior demanda (ROHINI; SATYANARAYANA; SRIKANTAN, 2016).

Outra tecnologia são as ondas milimétricas (*millimeter Wave*, ou, *mmWave*). As *mmWaves* fazem com que os sinais se propaguem em altíssimas frequências, entre 25 e 60 GHz e, por serem muito menores, diminui a chance de colisão e destruição das ondas, fazendo com que muitos mais dispositivos possam operar na mesma faixa de frequência (NIU *et al.*, 2015).

Em relação à economia de energia, o *Ultra-Dense Small Cell Deployment* foi proposto. O *Ultra-Dense Small Cell Deployment* utiliza um algoritmo em que um grupo de dispositivos de baixa capacidade não precisa se conectar diretamente à uma antena, e sim se conectar com outros dispositivos semelhantes, formando um *cluster*. Dentro desse *cluster*, o dispositivo com maior capacidade, ou uma estação base será o “*cluster head*”, que fará a comunicação com a antena (HUANG; PSOUNIS, 2019).

Com essas tecnologias, estima-se que a 5G irá ter os seguintes valores: taxa de dados máxima de 20 GBps, 1ms de latência, densidade de conexão de 10⁶ dispositivos conectados por quilômetro quadrado, dispositivos utilizarão 100 vezes menos energia para se comunicar com as antenas (em relação ao 4G) (PARIKH; BASU, 2020).

3. REDES LPWAN

A Internet das Coisas (ou, *Internet of Things*, IoT) foi definida pela ITU, em 2015, como: “Uma infra-estrutura global para a sociedade da informação, possibilitando serviços avançados, interconectando coisas (físicas e virtuais) com base em tecnologias de informação e comunicação interoperáveis existentes e em evolução.” (ITU, 2015 apud RAY, 2018, p.3 - tradução nossa).

A arquitetura da Internet das Coisas pode ser um sistema físico, virtual ou ambos, que consiste em uma coleção de inúmeras coisas, como sensores, serviços, dispositivos, protocolos, camadas e até eletrodomésticos (RAY, 2018). Com todas essas possibilidades, é possível observar várias aplicações para a IoT, como as *smart cities*, *smart homes*, segurança entre outros (MEKKI *et al.*, 2019).

Por causa da grande quantidade de dispositivos, as aplicações de IoT têm requisitos peculiares. É necessário que os dispositivos possam se comunicar em distâncias grandes, com baixa taxa de dados, baixo consumo de energia e que tenham um bom custo-benefício (MEKKI *et al.*, 2019).

As tecnologias de comunicação via rádio de curto alcance, como ZigBee e Bluetooth, não são recomendadas para aplicações de longo alcance. As redes celulares, como a 3G e 4G, têm uma ótima cobertura, mas consomem muita energia. Então, para atingir os requerimentos das aplicações de IoT, emergiu uma nova comunicação sem fio: a *Low Power Wide Area Networks* (LPWAN) (MEKKI *et al.*, 2019).

As redes LPWAN têm alcance de 10 a 40 quilômetros em áreas rurais, e 5 a 10 quilômetros em zonas urbanas. Também é altamente eficiente em relação ao consumo energético, baterias comuns chegam a durar mais de dez anos. São acessíveis, com um *chipset* custando menos de 2 euros e uma conexão de um ano custando menos de 1 euro. Essas três características fazem desta tecnologia ser a melhor escolha para um projeto IoT desde que o sistema precise transmitir poucos dados (MEKKI *et al.*, 2019).

Por causa dos dispositivos da LPWAN terem pouca capacidade de processamento e pouca taxa para envio de bits, essa tecnologia ainda não suporta o IPv6. Um bom exemplo pode ser apresentado: Enquanto que a LoRaWAN tem um *Maximum Transmission Unit* (MTU) de aproximadamente 256 bytes, o IPv6 iria precisar de 1600 bytes para funcionar corretamente. Mas a *Internet Engineering Task Force* (IETF) está trabalhando em uma forma de resolver esse problema desde 2016 (AL-KASHOASH; KEMP, 2017).

Estima-se que 50 bilhões de dispositivos estarão conectados via rádio em 2020. Com o rápido crescimento da Internet das Coisas, a LPWAN tem-se popularizado. A SigFox, LoRa e NB-IoT são as três principais tecnologias que disputam o mercado de grandes aplicações para Internet das Coisas (MEKKI *et al.*, 2019).

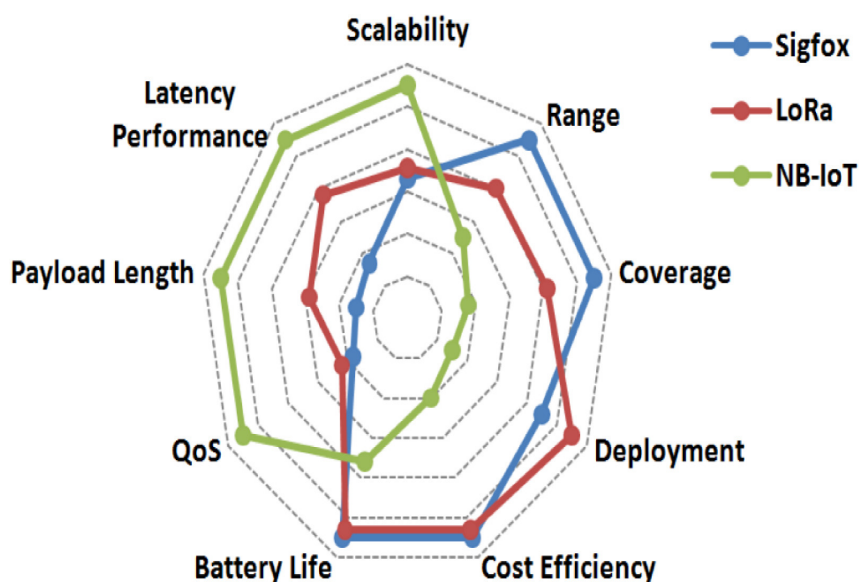
Uma comparação de preços entre SigFox, LoRa e NB-IoT é apresentada no Quadro 3, e uma comparação superficial das vantagens de cada tecnologia é explicitada na Figura 5.

Quadro 3 - Comparação de preços

	Custo do Espectro	Custo do Desenvolvimento	Custo do Dispositivo Final
SigFox	De Graça	>4000 euros por estação base	< 2 euros
LoRa	De Graça	>100 euros por <i>gateway</i> >4000 euros por estação base	3 - 5 euros
NB-IoT	> 0.5 euros/MHz	> 15000 euros por estação base	> 20 euros

Fonte: Adaptado de Mekki *et al.* (2019).

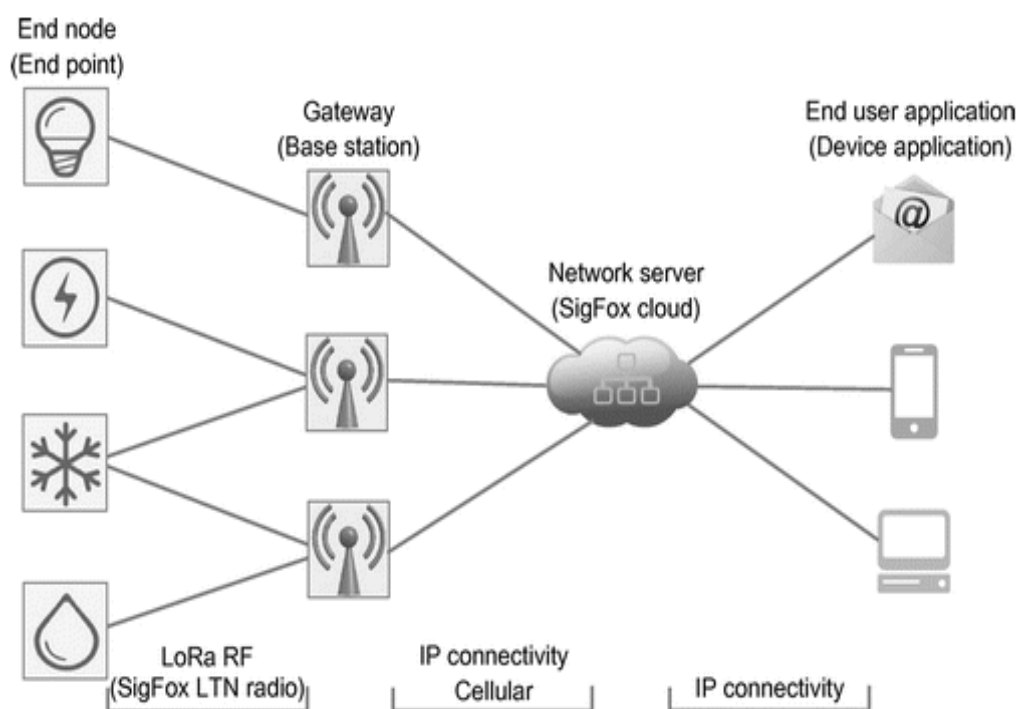
Figura 5 - Comparação de características



Fonte: Mekki *et al.* (2019, p. 5).

Pela Figura 5 e Quadro 3, pode-se perceber que as três tecnologias têm aplicações e casos de uso diferentes. Uma boa aplicação para a NB-IoT é o mercado de medição de energia elétrica, pois é importante ter baixa latência (*Latency Performance*) e uma alta qualidade de serviço (QoS, do inglês *Quality of Service*) para tomar decisões precisas e rápidas. Uma boa aplicação para LoRa e Sigfox é a rastreabilidade de produtos em grandes regiões, pois é muito necessário o baixo custo (já que é preciso muitas estações base e dispositivos), eficiência energética e cobertura para ter um sistema confiável e que pegue informações em várias localidades (MEKKI *et al.*, 2019). As arquiteturas da LoRaWAN e SigFox compartilham algumas semelhanças como são apresentadas na Figura 6. As duas redes consistem em quatro elementos: Ponto final (sensores), *Gateway*/Estação Base, servidor/*Cloud* e a aplicação do usuário (AL-KASHOASH; KEMP, 2017).

Figura 6 - Arquitetura SigFox e LoRa



Fonte: Al-Kashoash e Kemp (2017, p. 4).

A arquitetura representada na Figura 6 denomina-se “estrela”, pois ela tem um ponto central (servidor/*Cloud*) e várias pontas (sensores e aplicação do usuário). Vale ressaltar que os *gateways*/estações base são pontes transparentes. Os *end nodes* não conseguem comunicar entre si e, se comunicam, sem fio com a estação base (AL-KASHOASH; KEMP, 2017).

3.1 NB-IoT

Durante muitos anos, cada país tinha o seu tipo de comunicação para dispositivos móveis. Mas, com o crescente número de dispositivos e pessoas utilizando o serviço, tornou-se inviável fazer conversões entre as variadas tecnologias dos países. Com isso, em 1989, um grupo de pesquisa da *Conference of European Posts and Telegraphs* (CEPT) desenvolveu uma tecnologia padrão para a comunicação, chamada de *Global System for Mobile Communication* (GSM). Em 2017, essa tecnologia foi utilizada em mais de 200 países e mais de 1 bilhão de pessoas (BAKARE; EKANEM; ALLEN, 2017).

A *Narrow Band Internet of Things* (NB-IoT) é uma tecnologia que utiliza bandas pequenas para comunicar dispositivos da IoT. Criada em 2016, foi planejada para co-existir com a GSM e com a *Long Term Evolution* (LTE, também conhecida como 4G). A NB-IoT opera nas faixas de frequência licenciadas (700 MHz, 800 MHz e 900 MHz) com uma largura de banda de 200 KHz (MEKKI *et al.*, 2019).

A NB-IoT é baseada na LTE. As principais diferenças é que a primeira tem muito menos funcionalidades, mas melhora essas funcionalidades para os requisitos da Internet das Coisas. Por exemplo, no LTE é comum as antenas enviarem mensagens de *broadcast*, e isso

consome energia tanto da antena como de todos os receptores. Na NB-IoT, é raro ser enviada esse tipo de mensagem e, quando é enviada, é muito menor, reduzindo o custo energético (MEKKI *et al.*, 2019).

A NB-IoT utiliza a *Frequency Division Multiple Access* (FDMA) nas mensagens de *uplink*, com taxa máxima de 200 kbps e modulação *Quadrature Phase-Shift Keying* (QPSK) para mensagens de *downlink*, com taxa máxima de 20 kbps (MEKKI *et al.*, 2019).

Com uma mensagem de tamanho máximo de 1600 bytes, é possível que uma bateria comum dure mais de 10 anos enviando, em média, 200 bytes por dia (MEKKI *et al.*, 2019).

Estima-se que, com a NB-IoT, consiga-se conectar mais de 50 mil dispositivos conectados em um mesmo canal, com uma latência de 10 segundos no pior dos casos. Mas as principais vantagens são por reutilizar o *hardware* dos dispositivos que já conseguem se comunicar com a LTE e aproveitar a cobertura dessa mesma tecnologia, que é enorme (SINHA; YIQIAO; HWANG, 2017).

Por causa dessa tecnologia utilizar faixas de frequência licenciadas e ter comunicação síncrona, a NB-IoT tem os melhores índices de QoS entre as três tecnologias vistas neste capítulo. Mas essa vantagem tem um custo grande, leilões de faixas de banda normalmente saem por mais de 500 milhões de dólares por MHz (SINHA; YIQIAO; HWANG, 2017).

3.2 LoRaWAN

A camada física da LoRaWAN se chama LoRa (do inglês *Long Range*), que é uma técnica de modulação de banda proprietária (AL-KASHOASH; KEMP, 2017). A LoRa utiliza as bandas ISM (frequências reservadas para indústria, ciência e medicina do inglês *Industrial Scientific and Medical*) não licenciadas, por exemplo, 868 MHz na Europa, 915 MHz nas Américas, and 433 MHz na Ásia (MEKKI *et al.*, 2019).

A comunicação bidirecional é feita por meio da modulação *Chirp Spread Spectrum* (CSS) que envia um sinal de banda estreita por uma largura de banda de canal mais ampla. Resultando em um canal com menos ruídos, menos interferências e com menos chance de ser congestionado (MEKKI *et al.*, 2019).

A modulação CSS também faz com que o sinal seja resistente ao efeito Doppler. Fazendo, assim, com que dispositivos tenham uma alta performance ao comunicarem-se com objetos em movimento (MIKHAYLOV; PETÄJÄJÄRV; HÄNNINEN, 2016).

Outra vantagem da LoRa em cima de seus concorrentes é que a modulação tem seis *spreading factors* (SF, ou fatores de dispersão em português) ortogonais, que resulta em diferentes taxas de dados. Isso possibilita que vários sinais possam ser enviados ao mesmo tempo no mesmo canal sem perder qualidade da comunicação (MIKHAYLOV, PETÄJÄJÄRV; HÄNNINEN, 2016). A taxa de dados do LoRa é entre 300 bps e 50 kbps dependendo da largura do canal e do SF utilizado. Sendo que um SF mais alto possibilita um alcance maior, mas diminui a taxa de dados, e vice-versa (MEKKI *et al.*, 2019).

Cada mensagem pode ter até 243 bytes de tamanho útil, e uma mensagem enviada por um dispositivo é recebida por todas as estações base que estão no seu alcance. Isso cria uma redundância que acaba melhorando a porcentagem de mensagens recebidas com sucesso ao custo do preço de desenvolvimento, pois precisariam ser instaladas várias antenas em regiões

próximas. Essa redundância possibilita também a localização dos dispositivos com a técnica de localização baseada na diferença de hora de chegada (MEKKI *et al.*, 2019).

O formato da mensagem da camada MAC (*Media Access Control*, ou em português Controle de Acesso de Mídia) e da camada física da LoRaWAN é apresentado na Figura 7. Nela o PHDR é o cabeçalho da camada física, PHDR_CRC é a verificação por redundância do PHDR, MHDR é o cabeçalho da camada MAC, o FHDR é o cabeçalho do quadro, FPort é o campo da porta, MIC é o código de integridade da mensagem. O tamanho do preâmbulo é variável e depende da região e da modulação (AL-KASHOASH; KEMP, 2017).

Figura 7 - Formato da mensagem LoRaWAN

Octets: *	*	*	1	7 – 22	0 – 1	*	4	*
Preamble	PHDR	PHDR_CRC	MHDR	FHDR	FPort	Frame payload	MIC	CRC**

* Tamanho variável **Campo apenas para envio

Fonte: Al-Kashoash e Kemp (2017, p. 8).

As redes LoRaWAN podem ser públicas ou privadas. As redes privadas são proprietárias, ou seja, apenas indivíduos escolhidos podem utilizá-las. Enquanto que as públicas podem ser vistas como serviços que uma empresa de telefonia móvel pode prestar aos clientes (MEKKI *et al.*, 2019).

A segurança nas redes LoRaWAN são protegidas por duas chaves: a NwkSKey e a AppSKey, em que cada uma tem o tamanho de 128 bits. A primeira é uma chave de sessão que prova a integridade entre o usuário e a rede (como por exemplo, verificar a integridade da mensagem). A segunda chave entrega confiabilidade entre os *end-nodes* (sensores) e a rede, ela é utilizada para criptografar e descriptografar as mensagens com o algoritmo AES-128. (AL-KASHOASH; KEMP, 2017).

Por fim, a rede LoRaWAN tem três classes para *end-nodes* que têm requisitos e características diferentes. Classe A: dispositivos desta classe têm comunicação bidirecional com as estações base. Eles enviam suas mensagens quando necessário e podem receber mensagens da estação base em no máximo duas janelas. Por causa dessa comunicação especial, essa classe consome menos energia entre as três. Classe B: dispositivos desta classe podem abrir mais do que duas janelas de recebimento. Para abrir mais janelas, porém, é necessário um sinal da estação base. Isso faz com que a estação base saiba quando o dispositivo estará disponível para recebimento. Classe C: dispositivos desta classe estão quase sempre com suas janelas de recebimento abertas. Esses dispositivos só fecham a janela de recebimento para envio de mensagens. Esta classe consome mais energia das três, mas tem menos latência (MEKKI *et al.*, 2019).

3.3 SigFox

SigFox é uma rede LPWAN que oferece uma solução para a conectividade no mundo da Internet das Coisas baseada em suas tecnologias patenteadas. A SigFox, assim como a LoRaWAN, utiliza as bandas ISM não licenciadas para operar: 868 MHz na Europa, 915 MHz nas Américas, e 433 MHz na Ásia são as frequências reservadas (MEKKI *et al.*, 2019).

Os *end-nodes* da SigFox se conectam às estações base via modulação *Binary Phase-Shift Keying* (BPSK) em uma banda muito estreita, chamada de *Ultra Narrow Band* (UNB). Por causa da UNB, a comunicação via SigFox é feita de forma eficiente, com pouco ruído e pouco consumo de energia, mas tudo isso tendo uma vazão máxima de apenas 100 bps (MEKKI *et al.*, 2019).

Hoje, é possível ter comunicação bidirecional. Tanto do dispositivo (*end-node*) para a estação base (*uplink*), quanto da estação base para o dispositivo (*downlink*), o que durante um ponto foi uma desvantagem quando comparada aos concorrentes. Em comparação com a LoRaWAN, a SigFox oferece um alcance maior por estação base (AL-KASHOASH; KEMP; 2017; MEKKI *et al.*, 2019).

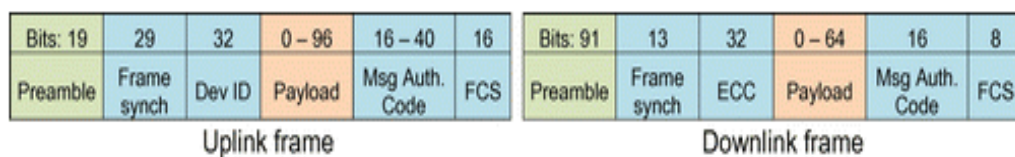
A SigFox, para impedir interferência tanto interna quanto externa na hora da comunicação, utiliza o *frequency hopping*. Essa técnica simplesmente troca o canal de comunicação quando for verificado que um outro dispositivo está usando o mesmo canal (AL-KASHOASH; KEMP, 2017).

O máximo de mensagens que um usuário pode enviar para a antena (*uplink*) é restrito a 140 mensagens, sendo que apenas 12 bytes são de *payload*. Enquanto que, de *downlink*, esse número é limitado a quatro mensagens com 8 bytes de carga útil (MEKKI *et al.*, 2019).

O sistema da Sigfox não utiliza ACKs (pacotes de *acknowledgements*) por causa da carga limitada de mensagens de *downlink*. Por causa disso, todas as mensagens enviadas por *uplink* são transmitidas várias vezes (3 vezes é o mais comum) utilizando *frequency hopping*, ou seja, cada mensagem é reenviada em um canal diferente (GOMEZ, 2019).

A Figura 8 apresenta a estrutura das mensagens da SigFox tanto para *uplink* quanto para *downlink*. O tamanho das mensagens de *uplink* (cabeçalho mais carga útil) podem variar de 14 a 29 bytes, enquanto para *downlink* varia de 20 a 28 bytes (AL-KASHOASH; KEMP, 2017).

Figura 8 - Formato do pacote do Sigfox



Fonte: Al-Kashoash e Kemp (2017, p. 6).

Olhando para o formato da mensagem de *uplink*, o quadro chamado *Frame Synch* é utilizado para sincronização, isso faz com que o recipiente consiga ler corretamente o resto da mensagem. Seguida pelo *Dev ID*, que é o ID do dispositivo. *Payload* é a carga útil que pode ser utilizada. O penúltimo quadro é *Message Authentication Code*, criado para poder autenticar, junto com o *Dev ID*, o *Payload*. Por último, vem o *Frame Check Sequence*, que com ele é possível verificar se não houve erros ou ruídos na transmissão. A única diferença da mensagem de *downlink*, em questão de quadros, é a troca do *Dev ID* pelo ECC. O ECC é chamado de *Error Correcting Code*, utilizado para corrigir erros de envios (GOMEZ, 2019).

Todo dispositivo que consegue se comunicar com a rede SigFox tem uma chave salva dentro do dispositivo pelo fabricante, a qual não pode ser trocada. Com isso, é autenticada

com 100% de precisão a mensagem, ou seja, verificar de que dispositivo veio. Mas as mensagens não são criptografadas (AL-KASHOASH; KEMP, 2017).

Todos os dispositivos que têm um modem da SigFox podem fazer comunicação unidirecional e bidirecional. Na primeira, o dispositivo envia a mensagem e não tem nenhum retorno, sendo assim, uma mensagem não verificada. Já, na mensagem bidirecional, logo após o envio da mensagem, o dispositivo abre uma janela de recebimento de uma mensagem *downlink*, que pode conter dados úteis ou não. Vale lembrar que, na SigFox, essa é a única forma de confirmar uma mensagem enviada, não existem ACKs para mensagens perdidas durante a transmissão (GOMEZ, 2019).

A Figura 9 apresenta de uma forma mais detalhada a arquitetura da rede SigFox. Nela, pode-se fazer um paralelo com a Figura 6 explicitada anteriormente. Na qual os *Devices* são os *end-nodes*, os sensores. As SigFox *Base Stations* constituem o segundo elemento, os *Gateways*. A *Cloud* é o servidor do sistema. Por último, a *Web-interface & API* é a *end-user interface*, ou o programa que o usuário poderá utilizar o sistema.

Todos os dispositivos se comunicam com as antenas (*Base Station*) via bandas ISM não licenciadas, como explicado neste capítulo. As antenas transmitem as mensagens para a SigFox *Cloud* (ou *backend*) via rede IP. Todas as mensagens são centralizadas no *backend* da SigFox, em que o desenvolvedor já tem uma interface para trabalhar. O *backend* também é responsável por fornecer essas mensagens a aplicações de terceiros via *Callbacks* ou API e, também, é responsável por filtrar as mensagens replicadas dos dispositivos. Todas as mensagens centralizadas na *Cloud* são salvas em dois bancos de dados, como *backup* (SIGFOX, 2017).

As antenas podem funcionar em diferentes zonas, conforme descrito anteriormente. O Quadro 4 ilustra as zonas de operação da SigFox e suas faixas de onda. Isto é importante levar em consideração sempre no momento do planejamento, pois é uma grande fonte de interferências locais (SIGFOX RADIO, 2021).

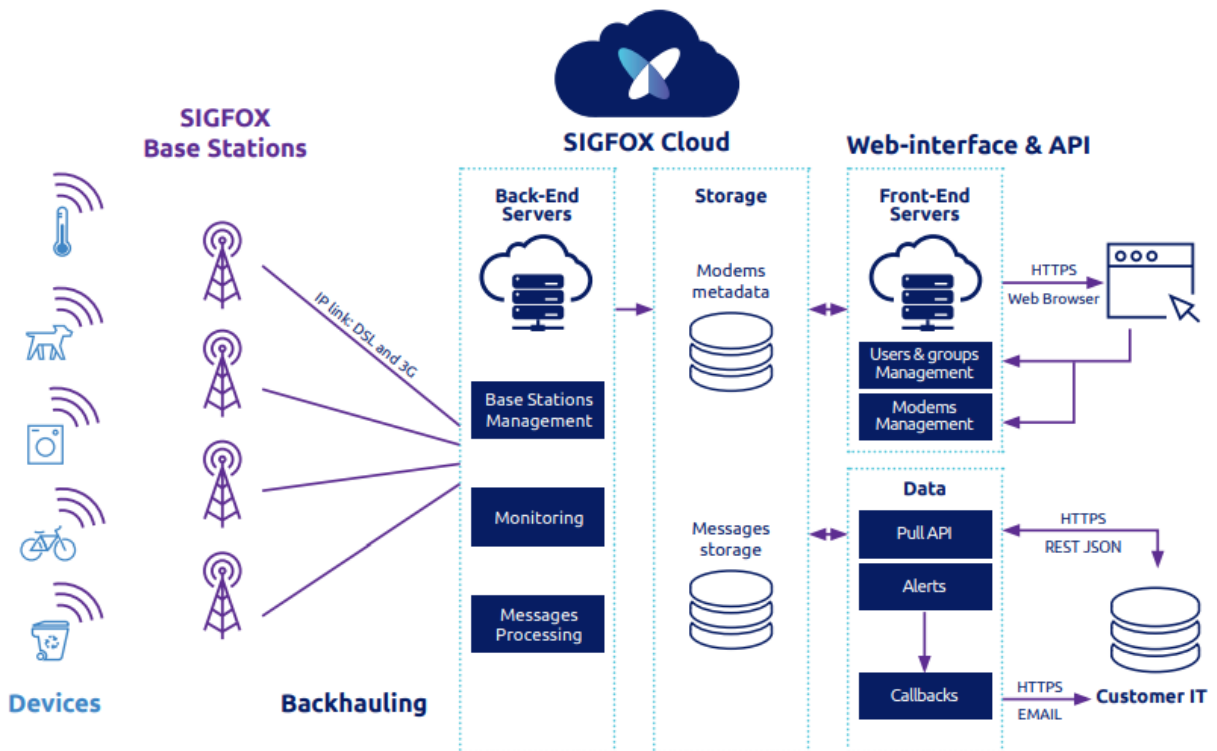
Cada coluna representa uma zona, indo da RC1 até a RC7, enquanto que cada linha representa um detalhe técnico. Nas duas primeiras, é apresentada a frequência da onda em *uplink* e *downlink* das regiões. Na RC2, a zona brasileira, as ondas precisam estar na faixa dos 902.2 MHz de *uplink* e 905.2 MHz de *downlink*. A terceira e a quarta linhas representam a velocidade de envio/recebimento de dados. Já a quinta e sexta linhas, representam coisas específicas das regiões, a quinta traz os decibéis por milliwatt (nível de potência em decibéis em relação ao nível de referência de um 1 mW), e a última linha representa uma funcionalidade específica da região. A RC2 é o envio de três mensagens iguais sempre (repetições) (SIGFOX RADIO, 2021).

Quadro 4 - Zonas da SigFox

	RC1	RC2	RC3	RC4	RC5	RC6	RC7
Uplink center frequency (MHz)	868.130	902.200	923.200	920.800	923.300	865.200	868.800
Downlink center frequency (MHz)	869.525	905.200	922.200	922.300	922.300	866.300	869.100
Uplink data rate (bit/s)	100	600	100	600	100	100	100
Downlink data rate (bit/s)	600	600	600	600	600	600	600
Sigfox recommended EIRP (dBm)	16	24	16	24	14	16	16
Specifics	Duty cycle 1% *	Frequency hopping **	Listen Before Talk ***	Frequency hopping **	Listen Before Talk ***		Duty cycle 1% *

Fonte: Sigfox Radio (2021, n.p.).

Figura 9 - Arquitetura da rede SigFox detalhada



Fonte: Sigfox (2017, n.p.).

Com o *backend* da SigFox, o desenvolvedor pode gerenciar os seus dispositivos e as mensagens de cada dispositivo no qual se pode verificar o *timestamp* das mensagens, os contratos dos dispositivos, localização do dispositivo (se for suportado pelo produto em questão) e o mais importante, pode-se configurar a comunicação do *backend* com alguma outra aplicação (SIGFOX, 2017).

O primeiro método de configuração é chamado de *Callback*. Com esse método é possível que toda mensagem enviada para a *Cloud* seja automaticamente enviada para a aplicação via uma requisição unidirecional HTTP (*Hypertext Transfer Protocol*). O segundo método é a API (*Application Programmable Interface*), que envia as mensagens recebidas via

requisições bidirecionais HTTP. Na API também é possível acessar os dados dos dispositivos, além de só os dados da mensagem (SIGFOX, 2017).

3.4 Sistema Embarcado para LPWAN

A forma mais simples de comunicar-se com uma rede LPWAN é por meio de um *modem* com comunicação com esse tipo de rede. Um desses *modems* é o LPWAN SigFox Node 868, que permite a comunicação com a rede SigFox (LAN CZECH IOT CLOUD, 2016).

Esse *modem* é totalmente compatível com microcontroladores que têm dimensões pequenas (24.31 x 14.97mm), controlado por comandos AT (comandos para se comunicar via GSM), conexão simples (apenas quatro fios), tem suporte para mensagens *uplink* (na frequência de 868.130MHz) e *downlink* (na frequência de 869.525MHz), baixo consumo de energia (65mA durante envio e 15mA durante recebimento), consome ainda menos energia no modo *sleep* (2uA), conexão UART, fonte de alimentação entre 1.8V e 3.6V (LAN CZECH IOT CLOUD, 2016).

Figura 10 - Modem da SigFox

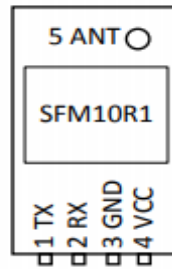


Fonte: Lan Czech Iot Cloud (2016, n.p.).

3.4.1 Pinagem

A Figura 11 apresenta a pinagem do *modem*. Na qual o pino 1 é a saída TX do UART, o pino 2 é a entrada RX UART, o pino 3 é o terra, o pino 4 é a fonte e o quinto e último pino é a antena para comunicação (LAN CZECH IOT CLOUD, 2016).

Figura 11 - Pinagem do *modem*

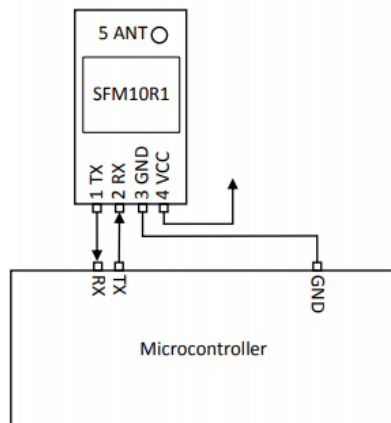


Pin No.	Pin Name	Description
1	TX	UART TX output.
2	RX	UART RX input.
3	GND	Power ground
4	VCC	Power VCC
5	ANT	Antenna input and output

Fonte: Lan Czach Iot Cloud (2016, n.p.).

A configuração UART para leitura/envio de mensagens é 9600 baud, 8 bits para dados, 1 bit para o sinal de finalização/início e sem bit de paridade (LAN CZECH IOT CLOUD, 2016).

Figura 12 - Conexão *modem* e microcontrolador



Fonte: Lan Czach Iot Ccloud (2016, n.p.).

A conexão com o microcontrolador não é complexa, apenas conectando a referência de ambos e depois o pino 1 do *modem* na entrada RX UART do microcontrolador e o pino 2 do *modem* se conecta com a saída TX UART do microcontrolador. Nesse caso, o microcontrolador envia comandos AT à antena e, assim, consegue fazer a comunicação com a rede SigFox (LAN CZECH IOT CLOUD, 2016).

4. SISTEMAS DISTRIBUÍDOS

A partir dos anos 1980, os computadores deixaram de ser objetos enormes e sem conexão, os processadores que, antes tinham barramento de dados de 8 bits, começaram a ser substituídos por processadores de 16 e até 32 bits. Outra mudança importante também foi a invenção das *Local Area Networks* (LAN), que permitiram a troca de informações entre centenas de computadores em locais próximos. Essas três mudanças contribuíram para o início dos sistemas distribuídos (STEEN, TANENBAUM, 2013).

Coulouris, Dollimore e Kindberg (2007, p. 15) definem sistemas distribuídos como: “Um sistema distribuído é aquele no qual os componentes localizados em computadores interligados em rede se comunicam e coordenam suas ações apenas passando mensagens”. Esse tipo de computação possui como pilares os paradigmas da programação concorrente e da programação paralela, em que dois ou mais processos trabalham juntos. A antiga ARPAnet pode ser considerada um sistema distribuído (MONTEIRO; RIVEIRO; BARRETO, 2018).

Os processadores passaram a ter mais de um núcleo de processamento, e as LANs cresceram e tornaram-se as *Wide Area Networks* (WAN), possibilitando a troca de informações em uma velocidade muito maior e entre computadores muito mais distantes. “Essas transformações contribuíram para o avanço do paralelismo e concorrência entre processos em um mesmo sistema.” (STEEN; TANENBAUM, 2013, p.15).

Todas essas inovações fizeram com que os sistemas distribuídos não fossem mais apenas uma possibilidade, e sim algo necessário, pois está em todos os lugares da computação moderna (STEEN; TANENBAUM, 2013).

De acordo com a definição, a computação distribuída consiste em duas principais características: a primeira é a coleção de elementos computacionais independentes, enquanto que a segunda é o único sistema coerente apresentado ao usuário (MONTEIRO; RIVEIRO; BARRETO, 2018).

Em relação à primeira característica, vale ressaltar que os elementos computacionais autônomos (ou nodos) podem variar desde computadores de altíssimo poder de processamento até sistemas mais limitados, como um sensor ou um teclado (STEEN, TANENBAUM, 2013). Apesar desses nodos trabalharem de forma independente, eles não podem ignorar os seus pares. Todos os dispositivos do sistema distribuído devem trabalhar em conjunto em prol de um objetivo comum (STEEN; TANENBAUM, 2013).

Um dos grandes desafios dessa característica são os diferentes *clocks* que cada dispositivo tem. Não se pode assumir que, em todos os casos em um sistema desse tipo, existirá um *clock* global, ou os *clocks* estão sincronizados. Algumas arquiteturas tentam resolver esse problema, como as arquiteturas cliente/servidor, *publisher/subscriber*, mestre/escravo ou a *peer-to-peer* (P2P) (STEEN; TANENBAUM, 2013) que são discutidas mais adiante.

Em qualquer um dos casos, os nodos do sistema devem estar sempre conectados possibilitando a troca de informações entre os mesmos (JIMÉNEZ; PUERTO; PAYÁ, 2017, p. 7). Essa organização de troca de informações é o segundo grande desafio desse tipo de sistema (STEEN; TANENBAUM, 2013).

A segunda característica da computação distribuída informa que o sistema deve aparecer ao usuário como um sistema único e coerente, ou seja, o usuário não deve perceber

que está lidando com diversos computadores. De forma simples, um sistema distribuído é coerente se ele se comporta de modo que o usuário espera independentemente de localização, como e quando o usuário interage com o sistema (STEEN; TANENBAUM, 2013).

Oferecer essa característica é o terceiro grande desafio desse sistema. Pois o usuário não pode distinguir onde as suas tarefas estão sendo realizadas, nem onde os dados estão sendo salvos. Isso gera um novo problema, pois nem sempre a rede de computadores inteira estará disponível ao usuário. Esse problema é muitas vezes resolvido com redundância, ou seja, informações são salvas em mais de um local, e mais de um nodo pode resolver a mesma tarefa (STEEN; TANENBAUM, 2013).

4.1 Arquiteturas de Sistemas Distribuídos

As arquiteturas dos sistemas distribuídos é a lógica na qual os nodos se comportam e se comunicam. O *design* e a adoção da arquitetura correta é um fator determinante no sucesso que o sistema terá em resolver o problema na qual será utilizado (STEEN; TANENBAUM, 2013).

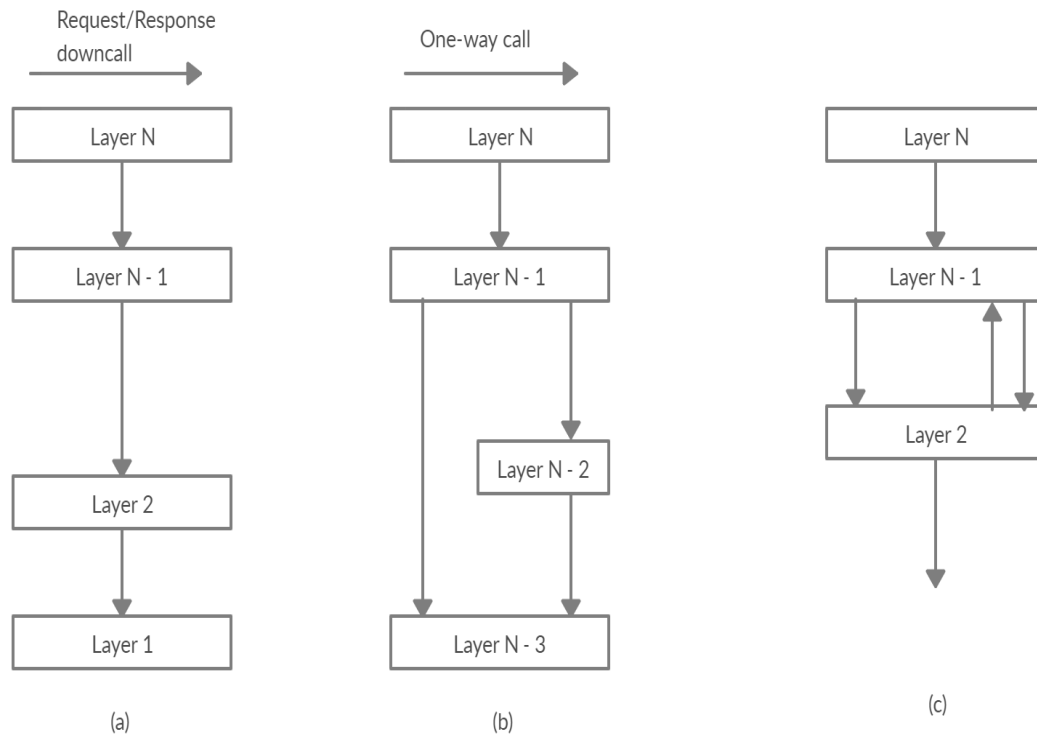
As arquiteturas são compostas por dois elementos, o componente e o conector. O primeiro é uma unidade com interfaces bem definidas e é substituível em seu ambiente. O segundo é um mecanismo que intermedia a comunicação, a cooperação ou a coordenação entre componentes, em outras palavras, possibilita a troca de informações entre o primeiro. É importante notar que o componente pode falhar ou ser substituído na maioria dos casos sem resultar em uma falha no sistema como um todo (STEEN; TANENBAUM, 2013).

Utilizando esses dois elementos, pode-se chegar a vários tipos de arquiteturas. Os mais importantes são: arquiteturas por camadas, arquiteturas centradas em recursos e arquiteturas baseadas em eventos (STEEN; TANENBAUM, 2013).

4.1.1 Arquitetura por Camadas

Nesse estilo de arquitetura, os componentes são organizados em camadas onde componentes de uma camada superior podem fazer chamadas para componentes de camadas abaixo e espera uma resposta. Raramente é feito chamadas de componentes de baixo para cima (JIMÉNEZ; PUERTO; PAYÁ, 2017).

Figura 13 - Os três casos comuns de chamadas



Fonte: Adaptado de Steen e Tanenbaum (2013).

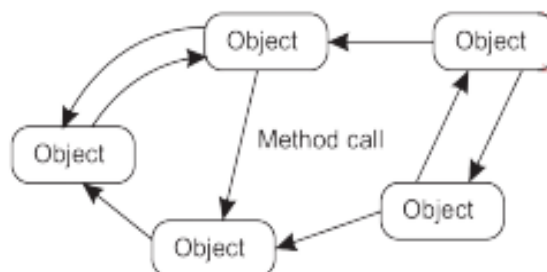
A Figura 13 mostra os três casos mais comuns de chamadas nesse estilo de arquitetura. Em (a) observa-se uma organização em camadas pura, onde sempre o componente superior faz chamadas para componentes abaixo, nunca o caminho contrário. Em (b) uma organização em camadas mixas, em que as chamadas também sempre vão de cima para baixo, mas não é obrigatório seguir essas chamadas, ou seja, a camada N-1 pode chamar direto a N-3, sem chamar a N-2. Em (c) pode-se ver uma organização em camadas com chamada de baixo para cima, onde os componentes inferiores têm a possibilidade de chamar os componentes superiores (STEEN; TANENBAUM, 2013).

Esse tipo de arquitetura é utilizado no protocolo *Transmission Control Protocol/Internet Protocol* (TCP/IP) nas camadas de transporte e de rede da internet. De modo que, os pacotes são transformados e trocados entre elas (JIMÉNEZ; PUERTO; PAYÁ, 2017).

4.1.2 Arquitetura Baseada em Recursos

A arquitetura baseada em recursos é mostrada na Figura 14, onde cada *Object* (ou, objeto) é o equivalente a um componente, e cada seta é o equivalente a um conector (STEEN; TANENBAUM, 2013).

Figura 14 - Arquitetura Baseada em Objetos



Fonte: Steen; Tanenbaum (2013, p.76).

A grande vantagem desta arquitetura é que os dados transmitidos entre objetos podem ser encapsulados (cada cápsula tem o estado do objeto e os métodos do mesmo). Todo objeto também é totalmente independente do ambiente, ou seja, cada objeto é facilmente substituído por outro objeto que tenha a mesma interface, também não precisa levar em consideração mais nada para fazer essa substituição (JIMÉNEZ; PUERTO; PAYÁ, 2017).

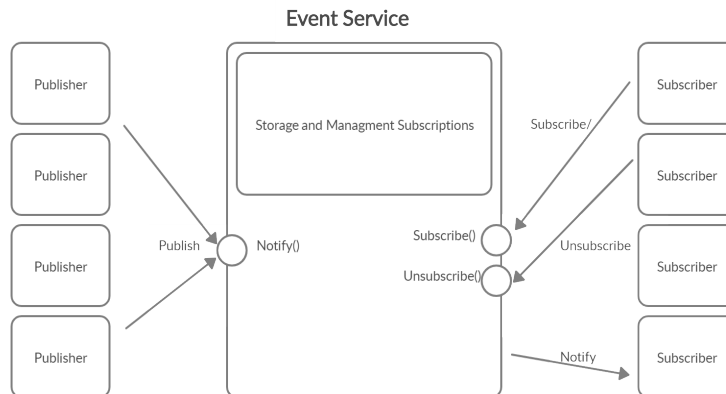
A interface desses objetos não precisa ficar no mesmo computador que os próprios objetos. A conexão pode ser feita através de um proxy, facilitando mais ainda uma possível substituição do objeto (STEEN; TANENBAUM, 2013).

Essas informações encapsuladas que são enviadas trocadas entre os objetos podem ser serviços. Um grande exemplo desse encapsulamento de serviços é o protocolo HTTP, onde a cápsula entre os objetos guarda os métodos (GET, PUT, POST, DELETE e OPTIONS), mas nesse caso em específico, não guarda o estado (STEEN; TANENBAUM, 2013).

4.1.4 Arquitetura Baseada em Eventos

Essa arquitetura é mais fácil de ser explicada com um exemplo, como a arquitetura *publisher/subscriber* (ou publicador/inscrito). Neste caso, os processos não conhecem uns aos outros (STEEN; TANENBAUM, 2013).

Figura 15 - Arquitetura *publisher/subscriber*



Fonte: Adaptado de Euster *et al* (2003).

A Figura 15 apresenta uma arquitetura de *publisher/subscriber*. Os processos só podem publicar uma notificação de ocorrência de um evento ou inscrever para um tipo específico de notificação. Toda nova publicação em uma notificação será enviada a todos os processos que se inscreveram em tal notificação. Todo o gerenciamento de notificações de envios e recebimentos de publicações é feito por uma entidade central (componente), que pode ser chamada de *Event Service* (EUGSTER *et al.*, 2003).

Toda comunicação é feita de forma assíncrona. As partes que estão interagindo não precisam estar ativas ao mesmo tempo. O publicador pode inserir novas informações no sistema enquanto o inscrito está *offline*. E o inscrito pode receber a notificação de algum evento enquanto o publicador está desconectado (EUGSTER *et. al*, 2003).

4.2 Aplicações

Todos os computadores ou dispositivos eletrônicos da atualidade são, provavelmente, apenas uma interface ou um ponto de um sistema distribuído muito maior. Esses sistemas vão desde objetos pequenos como microcontroladores e processadores multicore, até sistemas gigantes e complexos como o *Bitcoin* (STEEN; TANENBAUM, 2013).

4.2.1 Blockchain

Em 1989, Leslie Lamport submeteu um artigo chamado *The Part-Time Parliament* (ou em português, O Parlamento Meio Período define um algoritmo de consenso para tolerar faltas por parada onde um processo que apresenta falha por parada apenas deixa de processar e enviar mensagens (YAGA *et al.*, 2018).

Em 1991, uma cadeia de informações foi utilizada como sistema eletrônico para assinar digitalmente documentos. Esse método facilitou visualizar quais documentos assinados não foram alterados em momento nenhum (YAGA *et al.*, 2018).

Esses dois conceitos juntos e aplicados à dinheiro eletrônico em 2008 em um artigo chamado *Bitcoin: A Peer to Peer Electronic Cash System* (ou em português, *Bitcoin: um*

sistema de caixa eletrônico ponto a ponto). Este artigo foi publicado por um pseudônimo chamado Satoshi Nakamoto. Um ano depois, foi criada a rede *blockchain* do *Bitcoin* (YAGA *et al.*, 2018).

Leal (2017, p. 6) define *Blockchain* (ou em português, corrente de blocos) da seguinte maneira: “Um sistema distribuído de base de dados em log, mantido e gerido de forma compartilhada e descentralizada (através de uma rede peer-to-peer, P2P), na qual todos os participantes são responsáveis por armazenar e manter a base de dados.”

Um terceiro e importante conceito foi adicionado às *Blockchains*, o conceito de funções *hash*. *Hashing* (o verbo de utilizar uma função *hash*), é calcular uma saída única para uma determinada entrada sem que seja possível determinar quais variáveis foram colocadas na função. A Figura 16 apresenta exemplos de entrada e saída de uma função *hash* chamada SHA-256, observa-se que é impossível descobrir a entrada olhando para o resultado (YAGA *et al.*, 2018).

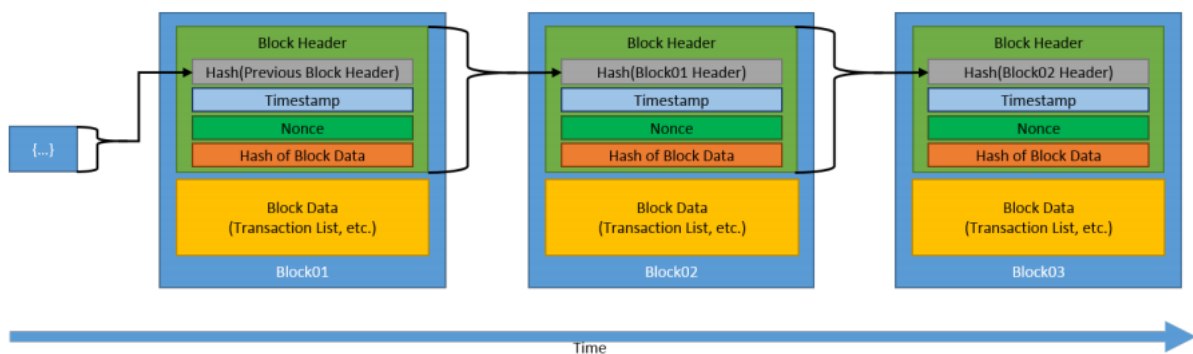
Figura 16 - Exemplos de entrada e saída de uma função *hash*

Input Text	SHA-256 Digest Value
1	0x6b86b273ff34fcea19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b
2	0xd4735e3a265e16eee03f59718b9b5d03019c07d8b6c51f90da3a666eec13ab35
Hello, World!	0xdf6021bb2bd5b0af676290809ec3a53191dd81c7f70a4b28688a362182986f

Fonte: Yaga *et al.* (2018, p.19).

O primeiro conceito é ilustrado na Figura 17, na qual é apresentada a composição de cada bloco. Observa-se o *Block Header*, que é o cabeçalho. O *Hash*, que é a combinação de todas as informações do bloco anterior em uma fórmula. O *Nonce* é o número aleatório que foi descoberto para que o bloco fosse criado. Em laranja, o *Hash* do bloco atual e em amarelo e por último, a informação guardada no bloco (YAGA *et al.*, 2018).

Figura 17 - Estrutura do *Blockchain*



Fonte: Yaga *et al.* (2018, p. 28).

Isso proporciona um grau de segurança maior do que bancos de dados comuns. Pois, se a informação em um dos blocos for alterada por um malfeitor, esse mesmo malfeitor teria que calcular novamente o *hash* de todos os blocos seguintes. No *Bitcoin*, por exemplo, em média cada bloco novo é criado em 10 minutos por causa da função *hash* utilizada. Isso faria com que se um criminoso alterasse a informação de um bloco, ele demoraria pelo menos 10 minutos em cada um dos próximos blocos para validar a sua alteração. Como a função *hash* é resistente a colisão, é improvável que sejam encontrados 2 *Nonces* ao mesmo tempo (YAGA *et al.*, 2018).

O segundo ponto importante da tecnologia é o consenso. Todos os usuários (nodos) da rede podem ter uma cópia da *Blockchain*. Enquanto a maioria dos nodos forem corretos, as cópias verdadeiras das informações da rede permanecerão inalteradas. No caso do *Bitcoin*, um nodo (sinônimo de minerador, nesse caso) sugere um bloco a ser adicionado na rede pelo PoW (*Proof of Work*, ou prova de trabalho). Os outros nodos verificam a integridade do bloco, e se for de comum acordo, esse bloco é adicionado à rede e o líder recebe uma recompensa (BENCIC; ZARKO, 2018).

O *Proof of Work* funciona da seguinte forma: o primeiro participante a resolver um quebra cabeça é eleito como líder. No *Bitcoin*, esse quebra cabeça é encontrar um número aleatório (*Nonce*) que, junto com as informações do bloco, formem uma saída na função *hash* que comece com uma sequência de 10 zeros, ou seja, o nodo tem que pegar todas as informações do bloco e ficar adicionando esse número até que a saída (como mostrada na Figura 13) comece com 10 zeros (BENCIC; ZARKO, 2018).

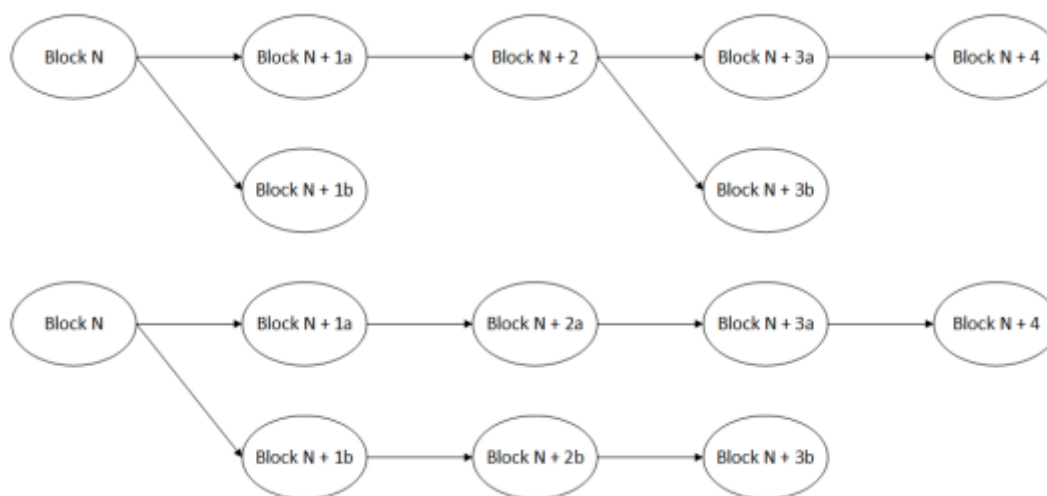
Depois que o bloco é descoberto, é necessário que os nodos validem se o bloco é confiável ou não. Mas, para isso, é necessário que os próprios nodos sejam confiáveis. É nesse momento que entra em ação o protocolo de consenso (BUCHMAN, 2016).

O mais famoso desses protocolos é o *Byzantine Fault Tolerance* (BFT, ou tolerância à falhas bizantinas). O termo “bizantinas” do nome vem do antigo exército do império Bizantino, onde os generais precisavam coordenar seus ataques contra Roma. Esse problema leva em consideração que os generais só se comunicam via mensagens e existem generais traidores (BUCHMAN, 2016).

Todos os nodos podem validar se o bloco é verdadeiro ou não. Mas no início eles não sabem essa informação, pois a função *hash* não é fácil de reverter. Então, a partir do *Nonce* que o minerador diz ter encontrado, cada nodo calcula o *hash* para verificar essa informação. Depois de verificado, cada nodo terá a sua resposta (se a *hash* foi igual ou não). Todos os nodos trocam as suas conclusões e chegam a um consenso pela maioria da rede (BUCHMAN, 2016).

Um fato comum é a *blockchain* ser dividida, ou seja, a partir de um bloco, são gerados dois outros blocos e, a partir desses dois blocos, a rede continua dividida por um período de tempo. Esse acontecimento é chamado de *soft fork*. Esse problema é resolvido por meio do “caminho” que tem o maior número de blocos. *Soft forks* podem acontecer quando dois nodos são eleitos líderes ao mesmo tempo, pois nem toda a rede recebeu o novo bloco ainda (BENCIC; ZARKO, 2018).

Figura 18 - *Soft Fork*



Fonte: Bencic e Zarko (2018, p. 3).

A Figura 18 apresenta dois exemplos de *soft forks*. No primeiro, pode-se perceber como o problema pode ser resolvido rapidamente, causando pouco estrago na rede. Já no segundo exemplo, apenas quatro blocos depois o problema é resolvido. Isso é um problema pois todas as transações ou informações nos blocos descartados são perdidas para sempre (BENCIC; ZARKO, 2018).

Todas essas características são características gerais, ou seja, a maioria das *blockchains* seguem tais características. Mas as *blockchains* podem ser divididas em três grandes (principais) grupos: sem permissão, públicas com permissão e privadas com permissão. A primeira, qualquer nodo pode se juntar a rede e começar a ler e escrever. Já na segunda, apenas nodos confiáveis podem se juntar à rede, mas qualquer outro nodo (mesmo que não esteja na rede) pode ler informações. Na terceira, apenas nodos escolhidos podem entrar na rede e apenas esses mesmos nodos podem escrever e ler dados (WUST; GERVAIS, 2018).

Muitos autores dizem que a tecnologia *blockchain* é imutável. Na verdade, essa tecnologia é resistente à violação e, por isso, o *blockchain* é muito utilizado e confiado em aplicações para fins transacionais. Um exemplo de como a tecnologia não é imutável foi discutido anteriormente, com o *soft fork*, onde um malfeitor com 51% da capacidade de processamento da rede poderia ter uma “rede paralela” que substituiria a “rede principal” (YAGA *et al.*, 2018).

Um ataque muito famoso às *blockchains* é chamado de ataque dos 51%, em que o atacante possui mais da metade dos recursos computacionais da rede. Tendo a maior parte dos recursos, o atacante consegue encontrar o *Nonce* mais rápido que os outros usuários fazendo assim um *soft fork*. Como ele produz blocos mais rapidamente, o seu caminho de blocos ficará maior, fazendo com que controle as informações que sejam salvas (YAGA *et al.*, 2018).

Blockchain é uma das tecnologias mais promissoras e que está com uma popularidade alta. A Figura 19 apresenta um conjunto de propostas, protótipos e provas de conceito que foram sugeridas na literatura para casos de uso de *blockchain* na Internet das Coisas (LEAL, 2017).

O primeiro projeto é uma plataforma de software na qual o usuário pode criar a sua aplicação com mensagens por *blockchain*. O *Chronicle* oferece uma gama de produtos para identificar dispositivos IoT, em que a identidade do dispositivo é guardada na *blockchain*. O terceiro projeto tem seus próprios contratos inteligentes para que máquinas consigam fazer transações financeiras com outras máquinas de forma totalmente segura. A empresa *Filament* desenvolveu um sistema que permite que *hardware* e *software* criem redes *mesh* automaticamente. A *Guardtime* tem um sistema que faz autenticação em larga escala sem uma entidade central. A IBM tem uma *blockchain* onde seus usuários podem replicar a sua própria rede nela para validarem as transações por meio de contratos inteligentes. A *Modum*, uma *startup* alemã, desenvolve dispositivos IoT que gravam variáveis ambientais de produtos para a rede de transporte de produtos. Já a *riddle&code* garante a identificação e autenticação de dispositivos IoT. A *Stampery* tem uma API para garantir a identificação de dispositivos utilizando a tecnologia do *Bitcoin* e da plataforma da *Ethereum*. Outra *startup* alemã, a *Slock.it* vende aplicações nas quais os dispositivos precisam ser identificados, realizar transações e participar de acordos. A *Tieron* permite que seus usuários gravem informações em uma *blockchain* para garantirem a integridade de seus dados. Por último, a *tilepay* permite que seus usuários cadastrem seus dispositivos IoT e terceiros acessem e controlem esses dados (LEAL, 2017).

A maioria das empresas dessa lista ainda é *startup*, apenas a IBM é uma multinacional, mas, mesmo assim, todos os projetos variam em grau de maturidade e de avanço com o projeto. Ou seja, o mercado ainda está se formando e ainda há a possibilidade de existirem novas empresas com novos projetos baseados em *blockchain* (LEAL, 2017).

Figura 19 - Exemplos de projetos reais utilizando *blockchain*

Categoria	Uso de Blockchain
Produto (plataformas de Desenvolvimento e aplicações)	Blockchain of Things (2016): - Mensagens seguras, armazenamento imutável, propriedades inteligentes, contratos inteligentes e transmissão de ativos.
	Chronicle (2017): - Identidade criptográfica para dispositivos físicos, proveniência segura de dados dos sensores e logs e automação de processos contratos inteligentes
	Eciotify (2017): - Autenticidade de dados, integridade de processamento e segurança de transação
	Filament (2016): - Resolução de endereços dos nós da rede de comunicação e execução de contratos inteligentes
	Guardtime (2017): - Autenticação e integridade de dados nos dispositivos IoT
	IBM (2017): - Replicação de dados de pares distribuídos e validação de transações com contratos inteligentes seguros

	Modum (2017): - Integridade de dados para transações de produtos físicos ao longo da <i>supply chain</i>
	riddle&code (2017): - Identificação de objetos físicos e autenticação digital.
	Slock (2017): - Identificação, pagamentos e acordos complexos entre dispositivos autônomos
	Stampery (2017): - Certificação de <i>firmware</i> e prova de integridade de dados entre dispositivos em rede mesh
	Tierion (2017): - Captura de dados de dispositivos conectados e registram dados e processos de negócio na Blockchain
	Tilepay (2017): - Autenticação, registro e publicação de dispositivos IoT, micropagamentos e chaves públicas

Fonte: Adaptado de Leal (2017).

4.2.1.1 Casos que o *Blockchain* não é Necessário

A tecnologia *blockchain* é muito cara em relação aos custos computacionais, seja em processamento, armazenamento ou transferência de dados. Apesar de a tecnologia ser a moda do momento, ela não é recomendada para resolver todos os problemas existentes na atualidade (PERLMAN, 2017).

Por causa da popularidade que o *Bitcoin* alcançou utilizando essa tecnologia, muitas pessoas querem criar qualquer tipo de aplicação utilizando *blockchain*. Mas o uso dessa tecnologia tem-se mostrado apenas útil em um nicho pequeno de mercado (PERLMAN, 2017).

O único fator realmente único desta tecnologia é que se pode chegar a um consenso entre milhares de nodos sem que nenhum deles seja controlado por uma entidade central. E se observar a maioria dos projetos propostos para essa tecnologia não utilizam essa característica em específico (PERLMAN, 2017).

De forma geral, essa tecnologia é melhor aplicada quando existem diversas entidades interagindo entre si e elas não confiam umas nas outras. E nenhuma entidade está disposta a confiar em uma terceira parte para controlar tudo. Por exemplo, um governo ou um banco (WUST; GERVAIS, 2018).

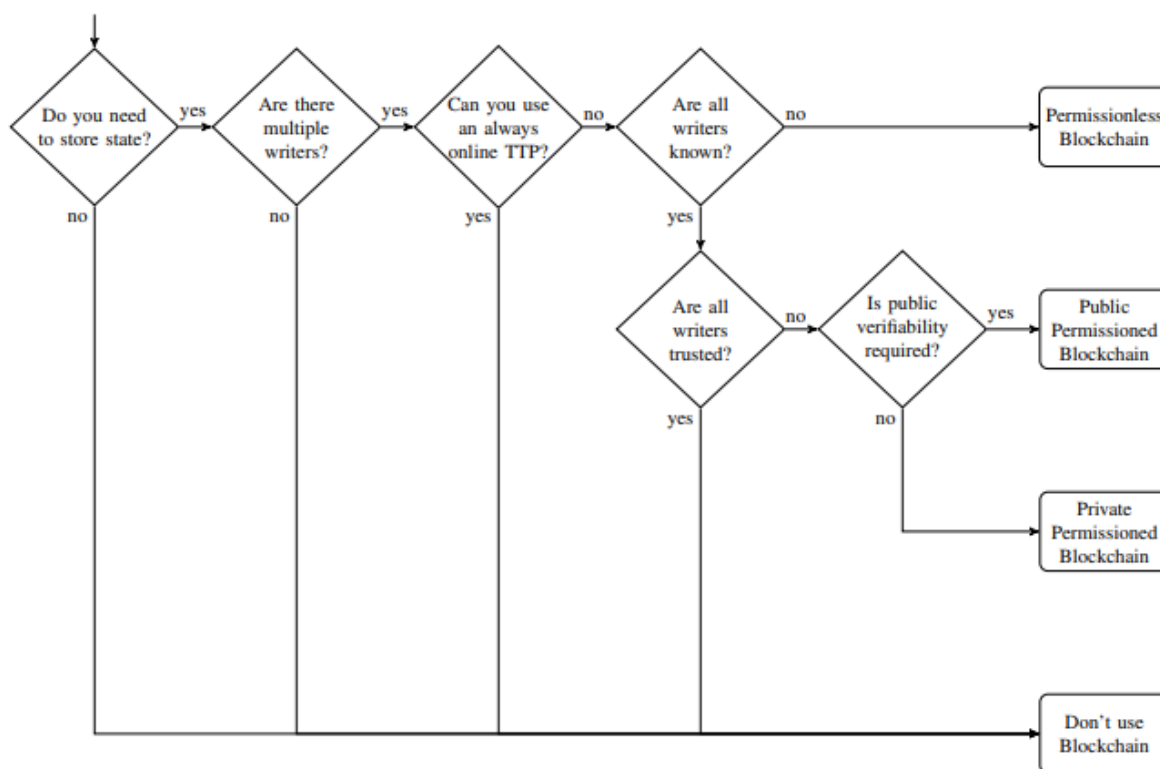
A Figura 20 simplifica a discussão se a *blockchain* será a tecnologia correta para resolver tal problema. Nela são consideradas múltiplas entidades que escrevem e lêem dados de um determinado banco de dados ou *blockchain* (WUST; GERVAIS, 2018).

O infográfico começa com uma pergunta: “Você precisa guardar o estado?”, onde o estado pode ser lido como dados. Se não é necessário guardar dados, não é necessário ter uma *blockchain*, pois ela foi criada para guardar dados de forma segura. Em seguida vem “Existem múltiplos escritores?”, se não existe, é mais barato ter um banco de dados comum (que é mais rápido), pois assim não precisa ter todo o trabalho de consenso que vem com uma *blockchain*. “Você sempre tem uma Terceira Entidade Confiável?”, se sim, apenas um banco de dados comum poderá ser gerido por essa entidade, já que ela é confiável. Se não existe esse

participante confiável é preciso se perguntar “Todos os escritores são conhecidos?”, se não, uma *blockchain* sem permissão é a melhor escolha, como o caso do *Bitcoin*. Se eles são conhecidos e são confiáveis, não precisam de algoritmos de consenso, então não é necessário uma *blockchain*. Mas se todos os escritores são conhecidos e não são confiáveis, então é necessário algum algoritmo de consenso, mas como eles são conhecidos, então é necessário uma *blockchain* com permissões. Se é pública ou não, vai variar se os nodos forem da rede e se podem ou não ler dados da rede (WUST; GERVAIS, 2018).

Por causa disso, desde os anos 1990, a *blockchain* não tinha sido utilizada por nenhuma aplicação muito grande. Por causa dessas características essa tecnologia encontrou o seu nicho de aplicação, que são as transferências financeiras, como no caso do *Bitcoin* e de tantas outras criptomoedas (PERLMAN, 2017).

Figura 20 - Infográfico sobre a decisão de ter ou não uma *blockchain*.



Fonte: Wust e Gervais (2018, p. 3).

Outro nicho de mercado que esta tecnologia está caminhando para ser necessária é o transporte de produtos. Nesse mercado vários atores precisam guardar dados de seus produtos, em que os mesmos atores precisam verificar se o produto está com uma qualidade boa ou não. Os clientes (nodos finais) também precisam poder verificar esses dados guardados, para decidir comprar ou não, ou seja, nesse caso, é necessário guardar dados, existem múltiplos escritores, não é sempre possível ter uma terceira entidade confiável e todos os escritores serão conhecidos (fabricantes, transportadores são exemplos). Como nodos de fora precisam ler (clientes), define-se que, para esse caso em específico, uma *blockchain* pública e com permissão faz muito sentido (WUST; GERVAIS, 2018).

4.2.2 Tangle-IOTA

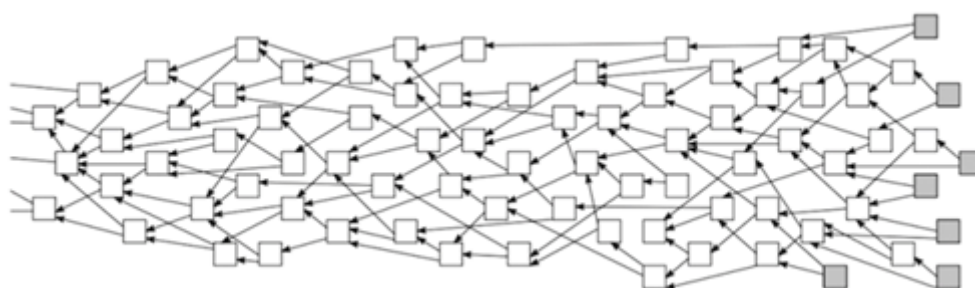
O *Tangle* foi conceitualizado em um trabalho conjunto de quatro pessoas em 2015. David Sønstebø, Sergey Ivanchev, Dominik Schiener e Dr. Serguei Popov criaram o conceito dessa tecnologia. Popov, em 2016, escreveu o *paper* chamado *The Tangle* que mostrou ao mundo como essa “nova *blockchain*” iria funcionar (BIRADAR; DYVIA, 2018).

O sistema montado pelo *Bitcoin* era e ainda é revolucionário, mas não se encaixa perfeitamente para um mundo da internet das coisas. Um dos problemas encontrados na *blockchain* foi a escalabilidade, em que cada transação tem que ser validada sequencialmente fazendo um gargalo na velocidade de todo o sistema. Já o *Tangle* foi desenhado desde o começo para ficar mais rápido à medida que o sistema cresce (BIRADAR; DYVIA, 2018).

Outro problema do sistema do *Bitcoin* são as taxas, em que toda transação é taxada pelo sistema em uma certa porcentagem para pagar o minerador que encontrou o *Nonce*. Em aplicações de IoT, é muito provável que essa taxa seja maior do que o valor sendo transferido, fazendo com que ela seja ilógica para esse cenário. Então, na IOTA, todas as transferências não têm taxa alguma (POPOV, 2016).

Em vez de ter uma rede uniforme e sequencial de blocos, essa nova rede tem um *Direct Acyclic Graph* (DAG, ou em português, Grafo Acíclico Direto) que é chamado de *Tangle* (POPOV, 2016). Trata-se de um DAG que é uma estrutura de dados que se move em apenas uma direção, nunca completando uma volta. Os blocos não precisam ter uma sequência em específico, ou seja, a rede pode ter várias pontas e vários caminhos para um bloco, mas se sair desse bloco, não há como voltar no mesmo, como mostra a Figura 21 (BIRADAR; DYVIA, 2018).

Figura 21 - Exemplo de um DAG



Fonte: Biradar e Dyvia (2018, n.p.).

Neste modelo, quando uma nova transação chega à rede, ela precisa aprovar (confirmar) duas outras transações. Essas aprovações são as arestas que conectam os blocos. Se de um bloco A até um B não existe uma aresta, mas existe um caminho de arestas e blocos, é possível falar que o bloco A aprova indiretamente o bloco B (POPOV, 2016).

A rede é composta por nodos. Esses nodos realizam e confirmam transações. Enquanto que os blocos podem ser chamados de *sites*, que nada mais são do que as próprias transações (POPOV, 2016).

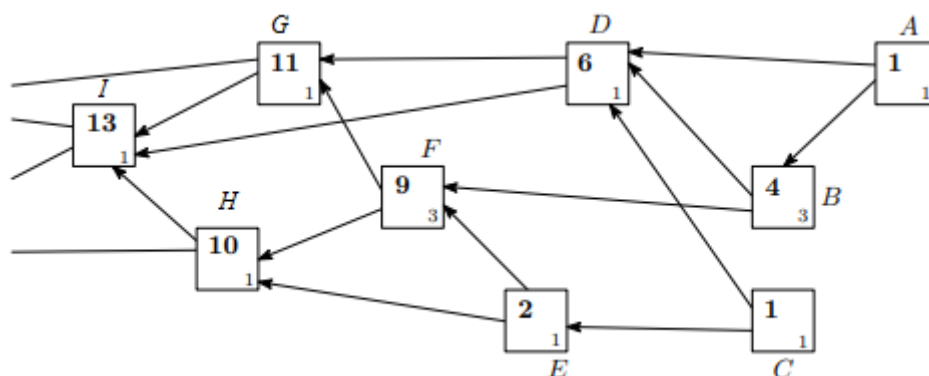
Para fazer uma nova transação, precisa confirmar outras duas, fazendo com que a confiabilidade do sistema aumente. Quando se diz confirmar uma transação, quer dizer verificar se essa transação está conflitando ou não com a rede (como por exemplo, o gasto duplo). Se for verificado que a transação está em conflito com a rede, o nodo simplesmente não irá aprová-la nem diretamente nem indiretamente (POPOV, 2016).

Como toda essa validação funciona de forma assíncrona e é bem possível que exista transações conflitantes dentro da rede, ou seja, os nodos não precisam entrar em consenso sobre qual transação tem o direito de estar na rede, todas as transações estarão na rede. (POPOV, 2016).

O que acontece com as transações conflitantes é que elas podem se tornar órfãs. A regra para decidir qual bloco entrará nesse estado é a seguinte: o nodo escolhe dois blocos para validar por meio de um *Tip Selection Algorithm* (TSA, ou em português, algoritmo de escolha de ponta) várias vezes e observa qual das duas transações conflitantes tem a maior chance de ser indiretamente aprovada pela ponta selecionada. Dessa forma, o nodo precisa encontrar as pontas com a maior probabilidade de serem confirmadas, para que a sua própria seja confirmada a seguir. As pontas que acabam por não ter confirmações, são as pontas órfãs (POPOV, 2016).

Um conceito importante para entender o TSA é o peso cumulativo de uma transação. Esse é valor do peso da própria transação mais os pesos das transações que a mesma confirmou direta ou indiretamente. Por exemplo o DAG da Figura 22, o peso da transação F será o peso dela mesma, mais o peso das transações que ela confirmou direta e indiretamente (A, B, C e E), logo o seu peso cumulativo é $1 + 1 + 1 + 3 + 3 = 9$. Vale notar também que os blocos A e C são chamados de pontas, pois eles não têm outras transações que os confirmem (POPOV, 2016).

Figura 22 - DAG com pesos das transações



Fonte: Adaptado de Popov (2016).

A escolha das pontas é algo muito importante caso o nodo queira que a sua transação seja validada. Existem vários algoritmos que podem fazer a escolha para o nodo. Mas o *paper* escrito por Popov estuda o caso de um algoritmo em específico, o *Markov Chain Monte Carlo* - MCMC (BIRADAR; DYVIA, 2018).

Esse algoritmo começa com o nodo colocando dois “caminhantes” em algum ponto do *Tangle*. Esse ponto não pode ser muito no início para não ficar muito complexo o cálculo, mas

também não pode ser muito no fim, para não ficar escolher qualquer ponta. Cada um desses caminhantes percorrerá um caminho até alguma ponta (blocos A e C) somando a quantidade de blocos que eles passaram e os pesos cumulativos de cada bloco, ou seja, se o caminhante está no bloco I (da Figura 20), ele precisa escolher se vai para o bloco G, bloco H ou bloco D. Ele irá escolher o bloco com o maior peso, pois ele é quem tem a maior probabilidade de ser confiável. Sem falar que assim o seu bloco terá um peso maior e terá a maior probabilidade de ser confirmado pelos próximos caminhantes (GU; HANA; POTOP-BUTUCARU, 2019).

Um dos problemas de tanta complexidade é o tamanho de todo o *Tangle*. Não é possível que dispositivos com pouca capacidade armazenem tantos dados (por exemplo, toda a rede do *Bitcoin* armazena mais do que 100GB). A IOTA resolve esse problema com eventuais *snapshots*, em que todos os balanços são salvos, e o histórico é removido, fazendo assim, com que a rede fique mais leve (BIRADAR; DYVIA, 2018).

Em média, é necessário checar 2^{68} possibilidades para conseguir encontrar o *Nonce* correto no caso do *Bitcoin*. Isso parece ser um número alto, ainda mais porque um computador comum precisa $O(N)$ (ordem de N) operações para encontrar o *Nonce*. Mas, pesquisas mostram que o computador quântico é muito bom em resolver problemas que dependem de tentativas e erros. Estima-se que esses novos computadores consigam encontrar o *Nonce* com $O(\sqrt{N})$ possibilidades. Ou seja, se as novas tecnologias de sistemas distribuídos não dificultarem os algoritmos, terá um grande número de ataques (POPOV, 2016).

Nisso a IOTA tem uma vantagem. Como a Prova de Trabalho que é feita no *Tangle* é muito pequena, um computador quântico tem muita pouca vantagem sobre computadores comuns. O trabalho realizado para achar no *Nonce* é tão baixo, que muitas vezes (se feito de forma assíncrona), o nodo terminará de realizar o *PoW* antes de realizar as outras tarefas para completar a transação (POPOV, 2016).

Uma grande vantagem que a IOTA tem sobre as outras criptomoedas é o *Masked Authenticated Messaging* (MAM). O MAM possibilita que os nodos possam enviar informações totalmente autenticadas e criptografadas. Fazendo com que apenas nodos autorizados possam ler e verificar essas informações (BIRADAR; DYVIA, 2018). Para que um dispositivo faça a comunicação via MAM, ele utiliza um ID de Canal no qual publicará todas as suas informações. Para receber esses dados, outro nodo precisa se inscrever nesse mesmo canal (PINJALA; SIVALINGAM, 2019).

Para fazer a verificação e autenticação de mensagens no MAM, é utilizado o *Merkle Tree Based Signature Scheme* (ou em português, Esquema de Assinatura Baseado na Árvore de Merkle). Em que o *root* (início) da árvore é o início do canal e cada mensagem tem o *root* para a próxima árvore (PINJALA; SIVALINGAM, 2019).

Cada mensagem é criptografada com o ID do canal, o *index* da chave que vai ser utilizada para assinar a mensagem. O *hash* resultante é assinado pela chave privada do nodo e então a mensagem é publicada. Assim, qualquer outro nodo com a chave pública poderá ler a mensagem (PINJALA; SIVALINGAM, 2019).

O MAM pode ser operado de três maneiras, dependendo de como os dados devem ser visíveis a terceiros. No modo público, o *root* da transação é o *root* da árvore, em que qualquer pessoa pode decifrar a mensagem com o *root* dela mesma. O segundo modo, chamado de privado, as mensagens são criptografadas, em que o *hash* do *root* da árvore é o endereço, ou

seja, só quem tem a chave pública do *root* pode decifrar e ler a mensagem. Por último, o modo restrito, existe também uma *SideKey* para criptografar a mensagem, o outro nodo precisa da *SideKey* e do *root* para decifrar a mensagem (PINJALA; SIVALINGAM, 2019).

5. SISTEMAS WEB

Quando a internet estava em seus primórdios, todas as páginas eram estáticas. Havia uma diferença entre usuários e programadores. Pois os primeiros apenas consumiam conteúdo, e os segundos produziam conteúdo (HARRIS; REA, 2019).

A partir dos anos 1994-1995, com a invenção dos blogs e *wikis*, o usuário passou a poder criar, compartilhar, colaborar e interagir de novas formas com o conteúdo, deixando de ser apenas um ator passivo. Essa nova internet foi chamada de *Web 2.0*, e os novos sites dinâmicos dessa nova *Web* foram chamados de sistemas web (HARRIS; REA, 2019).

No desenvolvimento de sistemas *web* é comum usar o termo “*stack*”, que representa todas as tecnologias para criar o sistema. Normalmente, uma aplicação desse tipo constitui-se de duas partes, o *front-end* e o *back-end*. O primeiro refere-se ao que o usuário vê (*client side*), como por exemplo a interface de usuário. O segundo é o trabalho em que o usuário não vê e mantém a aplicação funcionando e contendo informações dinâmicas (PORTER; YANG; XI, 2019).

De acordo com a pesquisa feita pelo *StackOverflow* em 2019, o *JavaScript* é a linguagem de programação mais popular entre desenvolvedores *web*. Isso explica a popularidade de uma *stack* chamada MERN, que tem todas as suas tecnologias em *JavaScript* (PORTER; YANG; XI, 2019).

MERN é uma sigla para os seus *frameworks*. O M é de MongoDB, o E é de Express, o R é de React e o N é de Node. O React é uma ferramenta para o *front-end*, enquanto que as outras três fazem o trabalho do *back-end* (PORTER; YANG; XI, 2019). Essas quatro tecnologias são apresentadas neste capítulo.

5.1 MongoDB

Esse é o banco de dados do MERN. Sendo um banco de dados não relacional, ele é mais rápido que bancos de dados relacionais, mas não consegue manter a integridade do mesmo jeito que seu concorrente. Por não ter relações, esse tipo de banco acaba escalando verticalmente, sendo ideal para aplicações IoT (PORTER; YANG; XI, 2019). As tabelas e linhas do SQL (*Structured Query Language*) são substituídos por coleções e documentos no NoSQL (NGUIEN, 2020)

A funcionalidade que é mais utilizada do MongoDB é a possibilidade de guardar dados dinâmicos em documentos do tipo BSON. Que significa que documentos que são da mesma coleção podem ter diferentes campos e pares chave-valor, enquanto que esses dados podem ser mudados sem nenhum tipo de restrição. Isso faz com que não sejam necessários dados estritamente estruturados, que é obrigatório em banco de dados relacionais, essa possibilidade aumenta a velocidade das operações realizadas no banco (NGUIEN, 2020).

Figura 23 - Exemplo de documento.



Fonte: Nguien (2020, p. 19).

A Figura 23 apresenta um exemplo de documento salvo em bancos não relacionais. O campo “_id” é o identificador único do objeto. Depois, vem um exemplo de campo chave-valor, em que a chave é “username”, e o valor é “123xyz”. Por último vem dois subdocumentos dentro desse documento, um chamado de “contact” e o outro chamado de “access”, sendo que cada subdocumento contém os seus próprios campos chave-valor. (NGUIEN, 2020).

Para fazer a conexão do MongoDB com o Sistema sendo desenvolvido, é muito comum usar o Mongoose. O Mongoose é um *Object Data Modeling* (ODM) escrito totalmente em *Javascript*. Ele permite que o desenvolvedor crie as suas estruturas dentro do banco de dados sem precisar aprender uma nova linguagem (NGUIEN, 2020).

Algumas empresas que utilizam a tecnologia MongoDB em 2019 são: Ebay, Cisco, Adobe, EA Sports, Google, Facebook, Nokia, SAP, GAP, e Verizon (PORTER; YANG; XI, 2019).

5.2 Express

O Express é um *framework* para o Node que é responsável pelas rotas da aplicação. Isso é possível pois o Express é construído em cima das requisições HTTP, como a GET, POST, DELETE e PUT (PORTER; YANG; XI, 2019).

O Express é chamado de *middleware* nas aplicações *web*. Esse *middleware* faz o gerenciamento de *cookies*, sessões, *logins*, parâmetros de URLs, *headers* das requisições e entre outras funcionalidades relacionadas às rotas (PORTER; YANG; XI, 2019).

O Express deixa a programação de aplicações *server-side* muito simples. Esse *framework* herda todas as propriedades do Node, como escalabilidade, facilidade, flexibilidade entre outras. Express é a *framework* mais famosa do Node, e quando alguém fala em trabalhar com Node, assume-se que vai utilizar o Express em conjunto (NGUYEN, 2020).

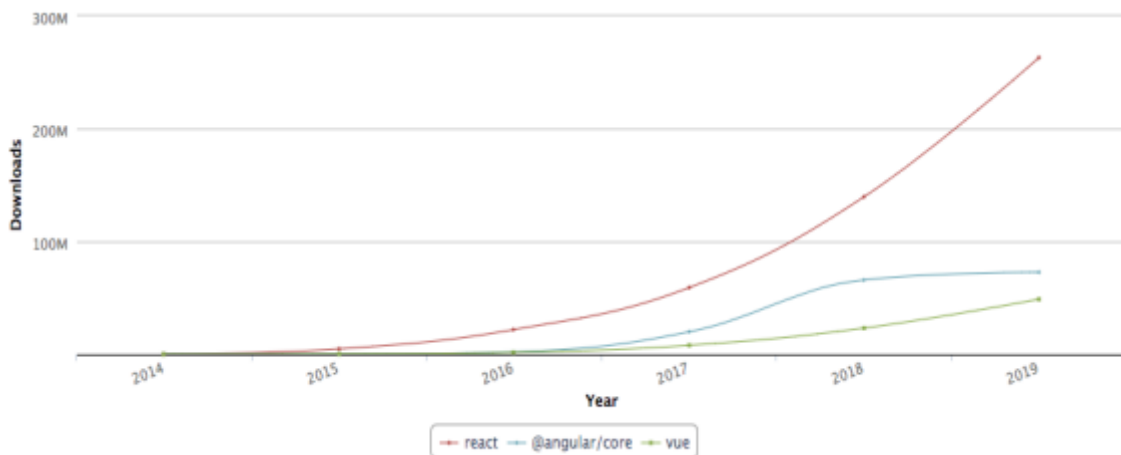
TJ Holowaychuk lançou o Express em 2010, mas hoje essa ferramenta é mantida pela Node *Foundation* e por desenvolvedores que contribuem para o código *open source* (NGUYEN, 2020).

Algumas empresas atuais que utilizam essa tecnologia são: Uber, Accenture e IBM (PORTER; YANG; XI, 2019).

5.3 React

React é o componente de *front-end* da MERN. Desenvolvido e mantido pelo Facebook. Uma das grandes vantagens do React sob os seus concorrentes é a facilidade de aprender a utilizar a ferramenta (PORTER; YANG; XI, 2019).

Figura 24 - Downloads de ferramentas para o *front-end*



Fonte: Nguien (2020, p. 23).

A Figura 24 apresenta a comparação do número de *downloads* do React e de outras duas ferramentas para o *front-end*, o Angular e o Vue. Percebe-se o crescimento exponencial do React, provando-se ser a principal ferramenta para desenvolvimento *front-end* da atualidade (NGUIEN, 2020).

O React funciona com componentes, em que cada pedaço da interface do usuário é desenvolvido como um componente individual que pode ser reutilizado em outra parte da interface ou até em outro projeto. O React, por ser desenvolvido em *Javascript*, herda da sua linguagem o poder de atualizar dados dinamicamente na página sem precisar recarregar (NGUIEN, 2020).

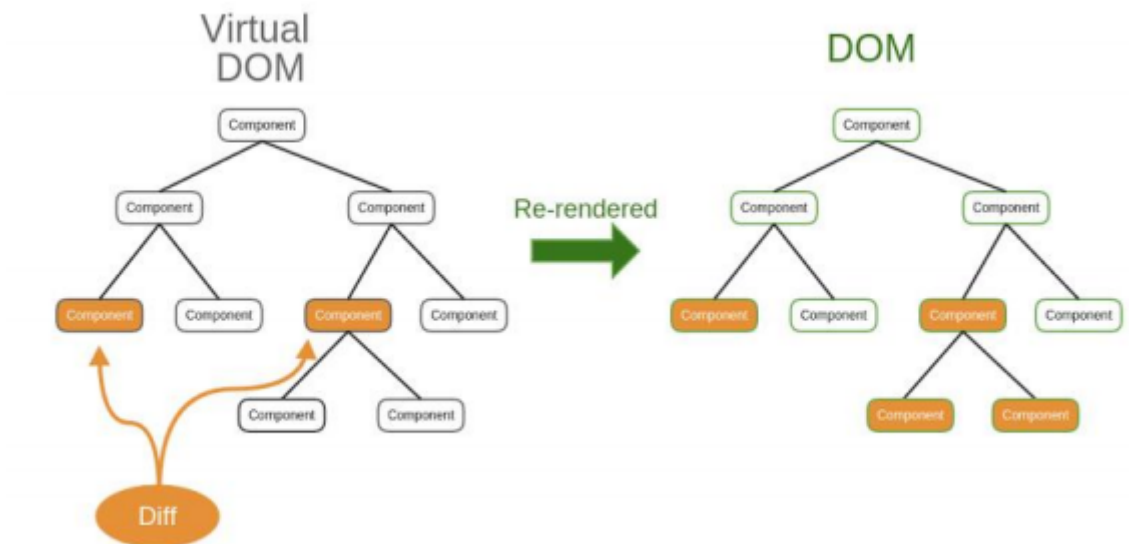
O React introduz o JSX, uma nova forma de estruturar os dados. Esse JSX é uma extensão do HTML comum com *JavaScript*, facilitando o tratamento dos dados para mostrar ao usuário (PORTER; YANG; XI, 2019).

Outra ideia inovadora do React é a *Virtual Document Object Model (Virtual DOM)*. A *Virtual DOM* é o principal motivo do React ser escalável. Aplicações *web* utilizam HTML para atualizar a DOM, e isso funciona para sites estáticos, mas pode comprometer muito a performance de sites dinâmicos com alta interação de usuário, pois toda vez que o usuário modificar algo, a página precisa ser carregada novamente (NGUYEN, 2020).

A *Virtual DOM* nada mais é que uma representação da DOM original, em que ela atualiza todas as informações antes de mandar a nova página para o *browser*. Toda vez que acontece uma mudança na interface, uma nova *Virtual DOM* é gerada e comparada com a

antiga. Com a comparação, o React consegue dizer quais partes da interface precisam ser alteradas, e apenas essas partes são recarregadas. Como resultado, é necessário muito menos poder computacional para modificar a interface, e muito menos carregamentos para o usuário (NGUYEN, 2020).

Figura 25 - Exemplo da *Virtual DOM*



Fonte: Nguyen (2020, p. 26).

A Figura 25 apresenta um exemplo da *Virtual DOM*. Os componentes laranjas foram modificados pelo usuário. Então, em vez de carregar a DOM inteira, o React consegue perceber apenas os seus componentes que sofreram alteração e carregar apenas o necessário (componentes laranjas à direita) (NGUYEN, 2020).

5.4 Node

O Node é uma ferramenta de *back-end*, o qual é um servidor assíncrono escrito em *Javascript* baseado na tecnologia Google V8 (que era do Google Chrome). Por utilizar o *Javascript*, o Node é executado em apenas um processo, mas isso, apesar de causar problemas de velocidade, reduz um bom número de *bugs* (PORTER; YANG; XI, 2019).

Toda vez que uma nova operação de *In/Out* (I/O) é necessária, em vez de ser aberta uma nova *thread* ou processo, o Node irá deixar a tarefa em espera, realizando outras tarefas enquanto espera a resposta desta (aqui é feito o trabalho assíncrono). Comparando com o PHP, o Node é muito mais confiável e rápido em todos os aspectos (PORTER; YANG; XI, 2019).

Quadro 5 - Comparação entre PHP e Node

Como o PHP lida com uma requisição com arquivo	Como o Node lida com uma requisição com arquivo
Envia a <i>task</i> para o sistema de arquivos do Sistema Operacional	Envia a <i>task</i> para o sistema de arquivos do Sistema Operacional
Espera enquanto o Sistema Operacional abre e lê o arquivo	Está pronto para receber uma nova requisição
Retorna o conteúdo ao cliente	Quando o Sistema Operacional abrir e ler o arquivo, o servidor retorna o conteúdo para o cliente
Está pronto para receber uma nova requisição	

Fonte: Adaptado de Nguyen (2020).

O Quadro 5 apresenta a principal diferença do Node, que é assíncrono, em relação ao PHP, que é síncrono. A única forma de uma ferramenta síncrona ser tão rápida quanto uma assíncrona, é por meio de paralelismo e concorrência entre *threads* e processos, mas isso acaba aumentando o nível de complexidade para o desenvolvedor (NGUYEN, 2020).

O *Node Package Manager* (NPM) é uma ferramenta de linha de comando que oferece ao programador acesso e instalação fáceis a mais de 350000 bibliotecas para o Node. Todos esses registros aceleram e facilitam o processo de criação de uma nova aplicação (NGUYEN, 2020).

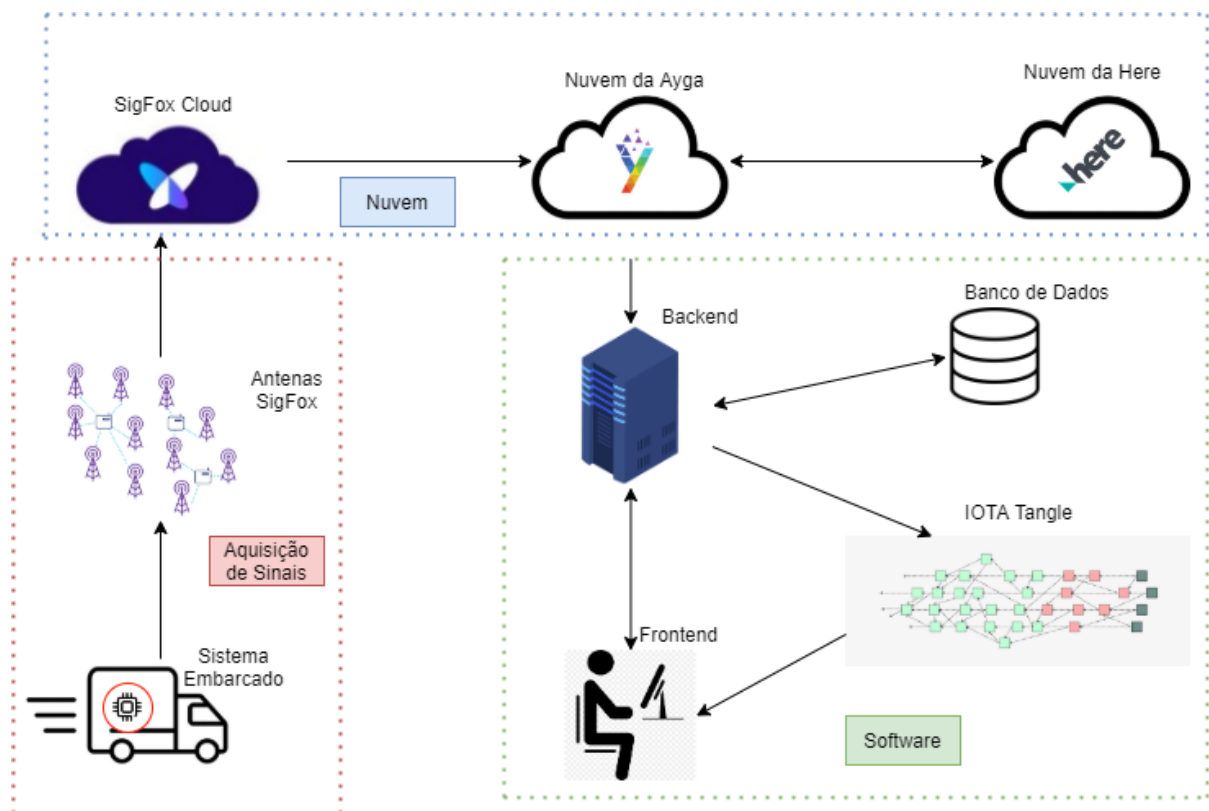
O Node é utilizado por grandes empresas, como o Walmart e o Paypal (PORTER; YANG; XI, 2019).

6. METODOLOGIA DESENVOLVIDA

Neste capítulo, é apresentado o modelo proposto para o sistema de rastreamento de mercadorias para prover mais segurança aos consumidores. Este capítulo descreve as etapas realizadas para a construção da prova de conceito baseada em IOTA e SigFox. O método apresentado abrange desde a aquisição dos sinais (dentro de um caminhão de carga, por exemplo) até a visualização e conferência dos atores sobre os produtos.

O fluxograma apresentado na Figura 26 representa de um nível maior de abstração a solução. A arquitetura pode ser agrupada em três grandes áreas, chamadas de Aquisição de Sinais, Nuvem e Software. A primeira representa a obtenção de uma variável analógica e uma variável digital e o envio de tais variáveis para a rede SigFox (circulado em vermelho). A segunda região, chamada de Nuvem, consiste no tratamento de tais dados pela Nuvem SigFox e pela Nuvem da Ayga (circulado em azul). Já a terceira etapa, o Software, representa o sistema *Web*, em que os dados são seguramente guardados, e os usuários poderão interagir com as medições (circulado em verde).

Figura 26 - Arquitetura da solução



Fonte: Elaborado pelo Autor.

As etapas da Figura 26 são descritas a seguir.

6.1 Requisitos do Sistema

Antes do desenvolvimento das três etapas descritas na Figura 26, foi necessário levantar os requisitos funcionais e não funcionais do sistema. A principal diferença é que o primeiro se refere ao que o sistema fará, enquanto que o segundo explica como o sistema realizará tais ações. Tais requisitos são listados nas duas subseções deste capítulo.

6.1.1 Requisitos Funcionais

- Um sistema deve facilitar acesso on-line a dados de sensores colocados no dispositivo de transporte de carga perecível, obtendo assim os dados ambientais e geolocalização proveniente deste;
- Deve ser permitido acessar os dados ambientais e geolocalização sem depender do sistema proposto, facilitando a auditabilidade;
- Os dados devem ser distribuídos em vários nós não proprietários;
- O sistema determina quem pode acessar um dado especificado após uma transação entre o solicitante e o servidor ou após uma declaração específica do servidor.
- O sistema permite consumir dados compartilhados pelos usuários para gerar novos serviços para os usuários.
- O sistema permite a qualquer usuário gerar serviços por meio dos dados obtidos por outros usuários: os provedores de serviços podem ser usuários indistinguívelmente simples ou organismos de controle confiáveis.
- O sistema valida os dados (por exemplo, localização) de um nó operativo que os solicita.

6.1.2 Requisitos não Funcionais

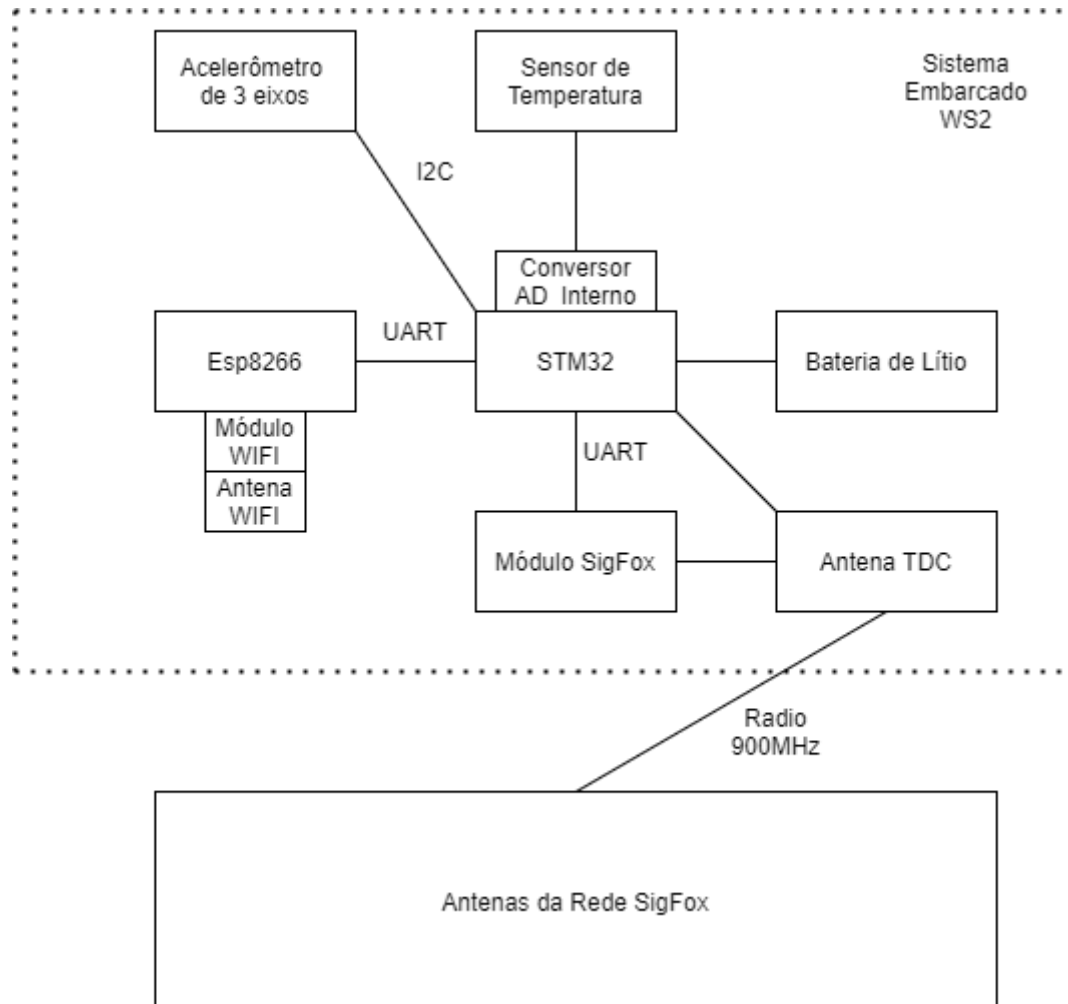
A fim de desenvolver essa prova de conceito, foi proposta uma arquitetura que sintetiza as ideias apresentadas, bem como favoreça o entendimento do produto e processo proposto. A Figura 27 sintetiza a arquitetura proposta que consiste em um sistema de transporte, contendo um sistema embarcado, conectado com sensores que coletam dados e enviam estes para a rede LPWAN (SigFox) - representado pela etapa “Aquisição de Sinais” - que, por sua vez, envia para um servidor que realiza e faz a integração - etapa: “Nuvem” - realiza a prova de trabalho necessária para anexar as informações no IOTA/Tangle, enviando os dados para o FullNode IOTA - etapa: “Software”.

6.2 Aquisição de Sinais

O primeiro passo da solução é a aquisição dos sinais. Esta etapa consiste em conseguir transformar grandezas físicas em sinais elétricos para que possam ser tratados e lidos por

diferentes computadores. Pelo presente trabalho ser uma prova de conceito, foram obtidas duas variáveis. A primeira é a localização, e a segunda refere-se à temperatura ambiente. A Figura 27 apresenta de forma mais detalhada essa seção de aquisição de sinais.

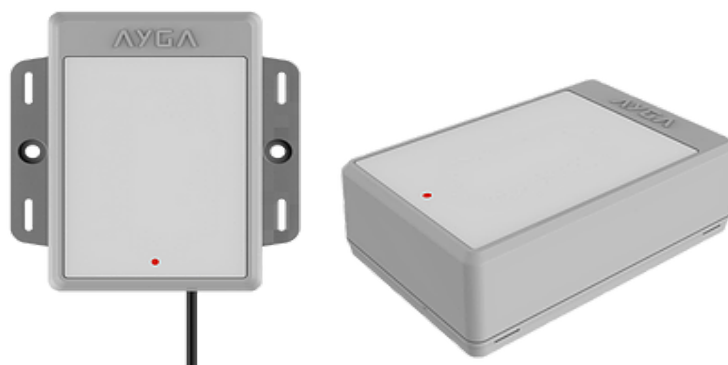
Figura 27 - Arquitetura da aquisição de sinais



Fonte: Elaborado pelo Autor.

Como sistema embarcado para o projeto, foi utilizado o WS2 da empresa Ayga (<https://www.ayga.com.br/>). Essa placa tem como objetivo monitorar a temperatura, localização e movimento de qualquer produto seja *indoor* ou *outdoor* sem ter necessariamente uma conexão física entre a placa e o produto com uma bateria interna que dura até 10 anos. O tamanho total da placa é de 89 mm x 59 mm x 28 mm. A Figura 28 apresenta o dispositivo.

Figura 28 - Dispositivo WS2



Fonte: Ayga (2021, n.p.).

A placa tem como microcontrolador o STM32 que funciona como “cérebro” do sistema embarcado. Esse microcontrolador opera com tensão entre 1.8 e 3.6V e tem um *clock* operacionalizado por um cristal que varia entre 4 a 26MHz. Com três ADC (*Analogic Digital Converter*) internos, 140 pinos I/O, duas entradas UARTs (7.5Mbit/s) e quatro entradas UARTs, duas entradas USB 2.0, ele é utilizado para centralizar todas as informações do sistema embarcado. Como processador, é utilizado o Arm Cortex-M3 com barramento de 32 bits e arquitetura RISC.

Conectado ao conversor AD interno do STM32 tem um Sensor de Temperatura NTC passivo de alta precisão. Este sensor consegue converter a temperatura ambiente (tem como limites -20°C e 70°C) em uma tensão linear com uma precisão de 0.1°C. Esta tensão, por sua vez, é convertida em um sinal digital pelo conversor AD para ser utilizado em outros lugares do sistema embarcado.

Um ponto muito importante do *design* do sistema embarcado é como ele faz para descobrir a sua localização. Normalmente, são utilizados módulos GPS (Neo-6m, por exemplo) para conseguir a geolocalização via satélite. O problema do sinal GPS é que ele funciona muito mal (muitas das vezes nem funciona) em ambientes fechados, como no caso deste trabalho, dentro de um container de um caminhão. Outro grande problema também é o consumo de energia, que o módulo GPS consome mais que as outras alternativas.

A solução encontrada para esse problema foi a triangularização de sinais WI-FI (*Wireless Fidelity*). Utilizando um ESP8266 que já tem um módulo e antena compatíveis com o protocolo 802.11 b/g/n da IEEE, que pode ler sinais da faixa dos 2.4GHz (velocidade de até 71 Mbps). Pode-se utilizar os protocolos HTTP/UDP/TCP, mas só existe suporte para o IPV4. Dentro das especificações, o ESP8266 escaneia por roteadores de WI FI próximos e escolhe uma quantidade de sinais específicos.

O componente não escolhe de forma aleatória tais sinais. Ele sempre pega os melhores sinais. A avaliação de quais sinais utilizar é feita com as seguintes variáveis: intensidade do sinal, sendo que os mais próximos ao dispositivo tem maior intensidade, exclui sinais com nomes de rede de celular e ou acesso remoto mobile, pois, como são móveis, podem prejudicar o cálculo, logo dá preferência para redes com SSID (*Service Set Identifier*) visíveis e também por MAC global.

Fazendo essa avaliação, são obtidos os MACs das redes - 3 ou 5 MACs, conforme configuração e precisão desejadas. O MAC de um roteador ou qualquer dispositivo que consiga se conectar a redes é um endereço único da placa de rede. Esse endereço é feito pelo fabricante da placa de rede e gravado direto no *hardware*, facilitando, assim, o controle de acesso de computadores a redes, como o nome sugere.

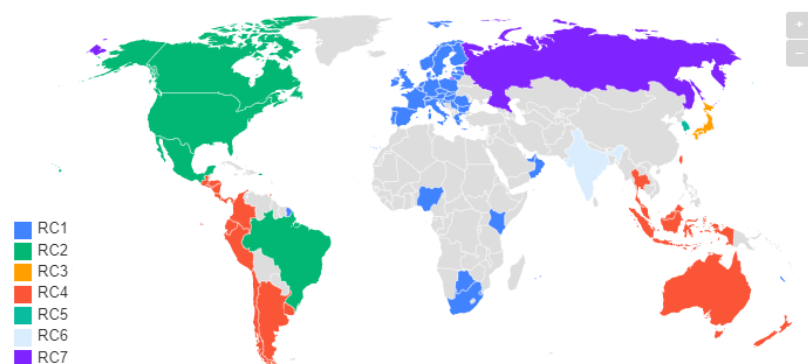
Depois de conseguir os MACs próximos e também a temperatura, tais dados vão como *payload* para a rede SigFox. Como visto nos capítulos anteriores, o tamanho da mensagem permitida pela SigFox é de apenas 12 bytes de carga útil, então, é preciso de mais de uma mensagem para enviar todos os dados necessários. Por exemplo, dentro de 12 bytes cabem apenas 2 MACs, logo é necessário mais de uma mensagem para que se envie os dados necessários.

Para realizar esse envio, primeiro é calculado o tamanho dos dados do *payload* dentro do próprio microcontrolador (para verificar se é necessário mais de um pacote, por exemplo) e, depois, enviado ao módulo SigFox. O próprio módulo já cria os *headers* do pacote e insere o *payload* para ser enviado. Muitas coisas que foram vistas em capítulos anteriores para o envio do pacote por meio da rede SigFox o módulo já deixa pronto. Por padrão, uma mesma mensagem é enviada três vezes para ter mais chances de chegar a uma antena.

O último componente do sistema embarcado é a antena. Se não o mais importante, a antena é uma das mais importantes peças dentro de provas de conceito como essa. Esse componente é responsável por converter os sinais digitais em ondas eletromagnéticas e vice-versa.

A antena foi desenvolvida do zero pela própria Ayga, pode se comunicar com as zonas RC1, RC2 e RC4 da SigFox e contém capacidades vistas nos capítulos anteriores. Como visto na Figura 29, essa placa pode comunicar-se em vários países diferentes.

Figura 29 - Zonas SigFox



Fonte: Sigfox Radio (2021, n.p.).

A antena utilizada é omnidirecional. Então o sinal é enviado em 360° a partir da antena, ou seja, tendo a antena como centro, o sinal é enviado em todas as direções. Em contrapartida, antenas desse tipo acabam por ter um alcance menor, pois a sua potência é distribuída igualmente em todas as direções.

Outra característica importante da antena é a sua topologia. Nesse projeto é utilizada uma topologia F invertido. Essa topologia consiste em uma antena monopolo (um condutor

reto em forma de haste), mas em vez de ser perpendicular ao plano de aterramento, ela é paralela e aterrada e uma das extremidades. Esse tipo de topologia traz duas vantagens: a primeira é que a antena é mais curta e compacta, enquanto que a segunda vantagem é que a combinação da impedância da antena pode ser controlada sem nenhum outro componente a mais.

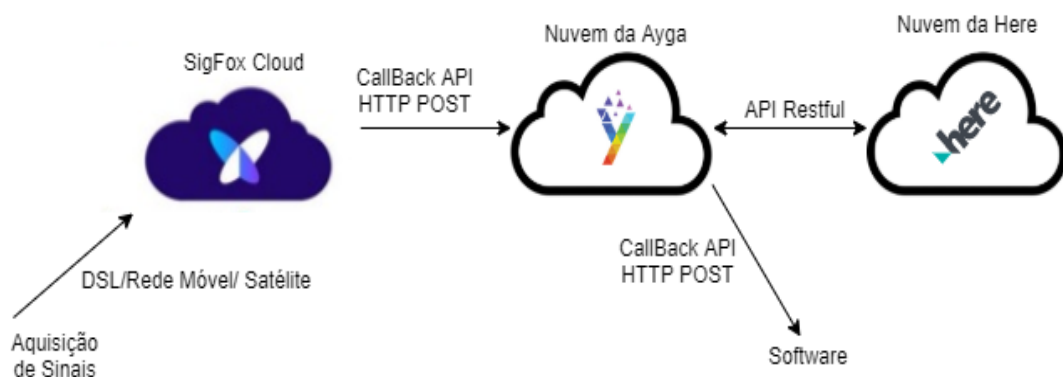
Depois da temperatura e dos MACs serem obtidos pelos periféricos (sensor de temperatura e ESP8266, respectivamente) e serem compactadas pelo microcontrolador (STM32), são inseridos em um pacote especial para a SigFox (módulo SigFox) e convertidos em ondas eletromagnéticas (antena), as quais são enviadas em uma frequência de aproximadamente 902 MHz (na região em que o Brasil está incluído) e recebidos por uma *base station* da SigFox.

Uma *Base Station* (sinônimo de antena, neste caso da SigFox) é responsável por transformar essas ondas de rádio em pacotes de dados para serem enviados via internet. Uma *Base Station* é formada por três “pedaços” de *hardware*: uma antena propriamente dita, que recebe a onda de rádio. Um LNA (*Low Noise Amplifier*, ou em português, amplificador de baixo ruído), que amplifica o sinal e filtra os ruídos. E, por último, um Ponto de Acesso, que envia o pacote pela internet. Normalmente é utilizado conexão DSL, mas, quando essa não está disponível, é utilizado 3G ou 4G. Quando nenhum deles está disponível, são utilizados satélites para realizar o envio. Todo esse processo é chamado de *backhaul*. Depois de convertido e verificadas as falhas, o pacote é enviado a *SigFox Cloud* pela internet.

6.3 Nuvem

Como mencionado na subseção anterior, após o pacote sair de uma *Base Station*, ele é recebido pela Nuvem da SigFox. Então, via HTTP POST, o pacote é enviado à Nuvem da Ayga e, por fim, enviado à próxima etapa via HTTP POST também. A Figura 30 ilustra tais etapas.

Figura 30 - Arquitetura da Nuvem



Fonte: Elaborado pelo Autor.

A primeira etapa desta subseção, a Nuvem da SigFox é onde é possível configurar a “ponte” entre o dispositivo que utiliza a rede SigFox e a aplicação final. Para o envio dos

pacotes para uma outra aplicação existem duas formas: a *Callback* API e a REST API. Nesta prova de conceito foi utilizada a primeira forma.

A *Callback* API é um meio de comunicação unidirecional (principal diferença em relação à REST API, a qual é bidirecional) que encaminha as mensagens de forma automática. Basta inserir um endereço (URL) que esse endereço receberá requisições HTTP. Depois de configurado o *callback*, não é preciso mais se preocupar com tais mensagens, pois todo pacote que chegar à Nuvem da SigFox será redirecionado à URL desejada via requisição HTTP POST.

Com a requisição completa à Nuvem da Ayga, é possível configurar algumas variáveis de leitura do *hardware* WS2 descrito anteriormente. Por exemplo, pode-se configurar o período de leitura de temperatura, fazendo assim que essa variável seja lida de acordo com esse período.

Com a identificação única de alguns roteadores próximos ao dispositivo, é possível triangularizar a posição do produto. A partir da nuvem da Ayga, os MACs obtidos pelo *hardware* são enviados a um serviço de terceiro, da empresa Here (<https://www.here.com/> para mais informações), onde são feitos os cálculos para descobrir a localização, com precisão de até 50m. Esta solução tem um menor consumo da bateria e menos tempo de processamento, quando comparado a um módulo de GPS. Na nuvem, os MACs obtidos são convertidos em latitude e longitude para que o sistema mais tarde possa usar tais localizações.

Uma configuração muito importante também é relacionada à localização do dispositivo. Anteriormente, foi mencionado que o *hardware* obtém os MACs de roteadores WI-FI próximos, aqui na nuvem, pode-se selecionar a quantidade de endereços. Vale a pena lembrar que 2 MACs já ocupam os 12 bytes de carga útil da SigFox. Além de 2 MACs, pode-se colocar para obter a localização com 4 MACs ou 4 MACs mais RSSI (Received Signal Strength Indicator, ou em português, Indicador de Força do Sinal Recebido), que, além dos MACs, utiliza a força do sinal recebido.

O RSSI (*Received Signal Strength Indicator*) é muito útil para melhorar a precisão da localização do dispositivo. Sem esse sinal de força, apenas se obtém os quatro MACs próximos, então a posição do dispositivo sempre fica “no meio” de tais endereços. Já, com o RSSI, é possível medir a distância do dispositivo de cada MAC, fazendo assim com que o sistema consiga uma posição mais precisa, pois não necessariamente o dispositivo estaria no meio de todos os MACs, ele pode estar mais próximo de um e mais longe de outros, por exemplo.

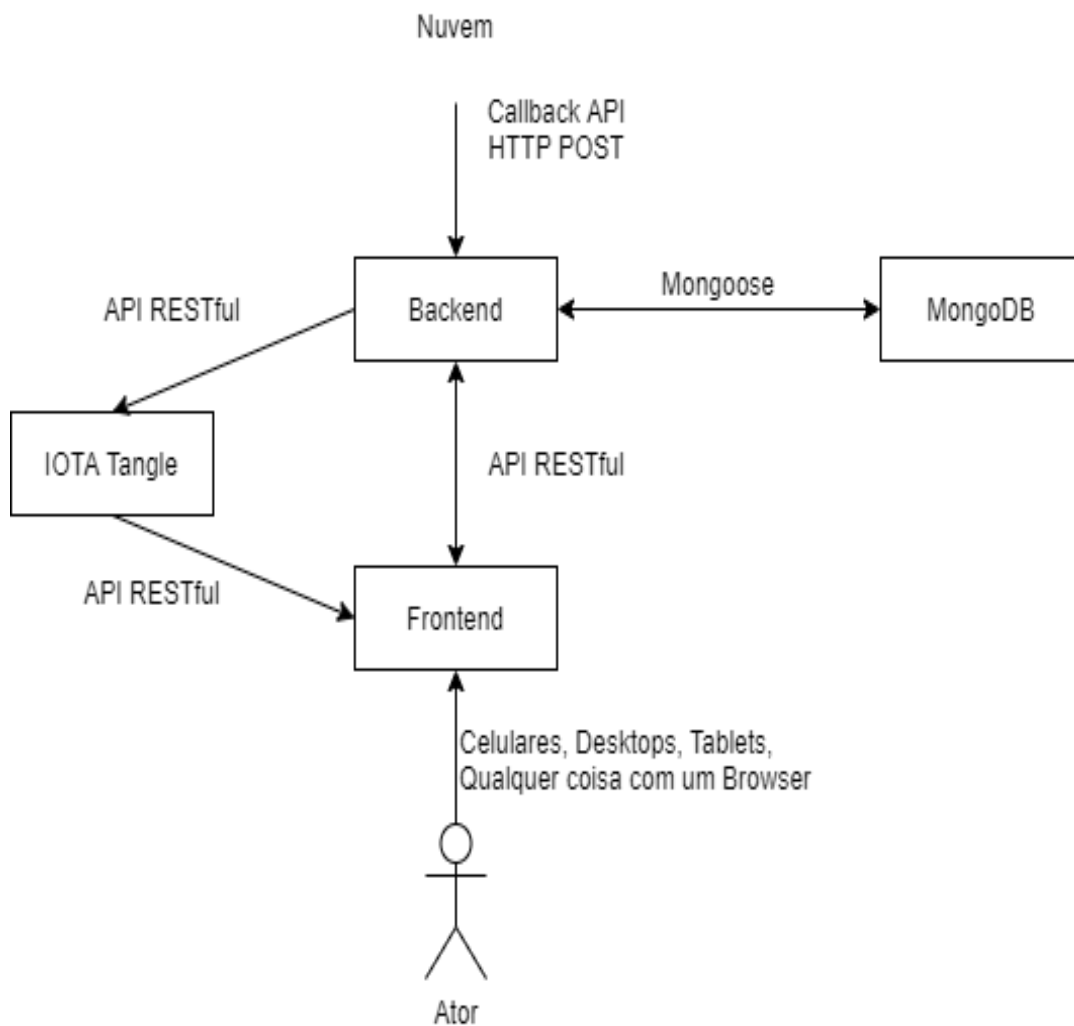
Percebe-se que a terceira configuração já utilizaria três pacotes SigFox para cada leitura necessária, mas aumenta e muito a precisão da localização. Esses dados obtidos para a localização (MAC e RSSI) são enviados à empresa HERE, que com o seu banco de dados devolve as coordenadas (latitude e longitude).

Dentro da Nuvem da Ayga ainda, o novo formato do pacote é criado, contendo o ID (identificador) do sistema embarcado, os valores obtidos juntos com o *timestamp* de quando eles foram lidos. Esse novo pacote é encaminhado via *Callback* API em uma requisição HTTP POST para a próxima etapa, o *Software*.

6.4 Software

As informações da nuvem chegam ao *backend* da aplicação já prontas para uso. Em que o *backend*, sem alterar em nada, envia o pacote direto ao Tangle da IOTA e registra ou não algo no banco de dados. Com isso, um usuário qualquer consegue acessar e comprovar tais dados de maneira fácil via o *frontend*, podendo acessar com qualquer dispositivo que tenha acesso à internet. A Figura 31 apresenta esse trâmite (arquitetura da etapa *Software*) e logo na sequência é visto com mais detalhes cada entidade.

Figura 31 - Arquitetura do Software



Fonte: Elaborado pelo Autor.

A primeira entidade desta etapa, o *backend* foi escrito em Node JS e Express. Esse componente é responsável por enviar os pacotes da Nuvem para o Tangle e também por fazer uma conexão com um banco de dados, o MongoDB. Dentro do corpo da requisição HTTP POST, recebido da Nuvem, contém um JSON com as informações necessárias. A Figura 32 apresenta um exemplo desse JSON.

Figura 32 - Exemplo de JSON enviado ao Tangle

```
[
  {
    "deviceUUID": "PROT-0202-202",
    "signals": [
      {
        "UUID": "temperature",
        "logs": [
          {
            "date": "2020-11-24T16:06:47.000Z",
            "value": 28
          },
          {
            "date": "2020-11-24T16:09:47.000Z",
            "value": 27
          },
          {
            "date": "2020-11-24T16:12:47.000Z",
            "value": 28
          }
        ]
      },
      {
        "UUID": "positionDOTS",
        "logs": [
          {
            "date": "2020-11-24T16:06:47.000Z",
            "value": {
              "lat": "-28.811856",
              "lng": "-49.385966",
              "radius": 250
            }
          },
          {
            "date": "2020-11-24T16:49:47.000Z",
            "value": {
              "lat": "-28.477765",
              "lng": "-49.032339",
              "radius": 250
            }
          },
          {
            "date": "2020-11-24T17:32:47.000Z",
            "value": {
              "lat": "-27.602122",
              "lng": "-48.627705",
              "radius": 250
            }
          }
        ]
      }
    ]
  }
]
```

Fonte: Elaborado pelo Autor.

Dentro desse JSON, é possível identificar informações momentâneas da viagem. Cada placa é responsável por obter os dados de uma única viagem. Pode-se entender da seguinte forma: um sistema embarcado fica dentro de um caminhão. Logo, esse dispositivo só obterá os dados de tal caminhão e de seus produtos que estão na sua caçamba. Por isso, cada placa identifica uma “viagem”, pois ela consegue obter os dados do movimento do caminhão desde a sua carga, até que os produtos sejam entregues. À medida que o tempo passa, a placa vai

pegando novas temperaturas e novas localizações, podendo assim, ir atualizando a viagem até o seu término.

Nesta etapa entra o motivo de ter dois bancos de dados dentro da aplicação. O primeiro, o Tangle da IOTA, é responsável por armazenar e deixar seguras as informações sobre as viagens em até 2 meses, que é quando acontece um *snapshot* da rede. Já o segundo, é utilizado MongoDB para guardar informações do usuário, como contas (nome, senha, entre outros) e informações sobre distribuidores e suas viagens.

Vale notar que o *backend* não tem nenhum tipo de interferência nas informações que chegam ao Tangle, ele serve apenas como intermediário. Outra coisa importante que o *backend* realiza é a verificação do ID da placa. Pode-se ver na Figura 32 um campo chamado “deviceUUID”, esse campo informa de qual placa esses dados estão chegando.

As Figuras 33, 34 e 35 apresentam os procedimentos feitos pelo código que insere os dados recebidos no *Tangle*. A Figura 33 ilustra o recebimento das informações por uma rota do *Express*, essas informações sempre estão no *body* (corpo) da requisição. Figura 34 externaliza essa mesma rota passando os dados para a função “*attachToTangle*”, que é responsável por enviar os dados ao *Tangle*. Por último, a Figura 35 mostra a função “*attachToTangle*” por completo.

Figura 33 - Inserção no *Tangle*

```
let device = req.body[0].deviceUUID;
let body = req.body[0];
let response = {}, info = {};
```



```
delete body['deviceUUID'];
```

Fonte: Elaborado pelo Autor

Figura 34 - Chamada da função para inserir no *Tangle*

```
info = await attachToTangle(body);
```

Fonte: Elaborado pelo Autor.

A função “*attachToTangle*” recebe dois parâmetros, os dados e um estado da IOTA. O primeiro parâmetro são os dados a serem enviados para a API, enquanto que o segundo representa as informações de um canal da IOTA. Em seguida, verifica se foi passado um estado, se for passado, é necessário utilizá-lo, pois quer dizer que são informações de uma viagem existente, e são necessárias que todas essas informações fiquem no mesmo canal para que possam ser lidas mais tarde. Caso não seja passado um estado, quer dizer que é uma nova viagem, então é necessário inicializar um estado (*Mam.init()*) e começar do zero. Dessa forma, todas as informações de cada viagem sempre estarão separadas em um canal da IOTA. Depois, os dados são convertidos para *trytes* (*asciiToTrytes()*) e criado uma mensagem da IOTA (*Mam.create()*). Na linha 133, é salvo o estado criado da IOTA e na linha 135 é adicionada essa mensagem criada no *Tangle*, passando como parâmetros o endereço da mensagem, o *payload* (conteúdo), o nível de segurança e a quantidade mínima de *proof of work* para ser realizada. Por fim, são retornadas as informações necessárias para serem salvas

no banco, caso seja necessário enviar novas informações ou visualizar os dados pelo *front-end*.

Figura 35 - Função para inserir no *Tangle*

```
119 async function attachToTangle(jsonData, mam_state = null) {
120   let config = await loadConfiguration();
121   let mamState;
122
123   if(!mam_state) {
124     mamState = Mam.init(config.mam_provider);
125     mamState = Mam.changeMode(mamState, config.mam_mode);
126   } else {
127     mamState = mam_state;
128   }
129
130   const trytes = asciiToTrytes(JSON.stringify(jsonData));
131   const message = Mam.create(mamState, trytes);
132
133   mamState = message.state;
134
135   let transactions = await Mam.attach(message.payload, message.address, 3, 14);
136
137   let info = {};
138   info.state = mamState;
139   info.root = message.root;
140
141   if (transactions.length == 0) {
142     throw new Error('erro ao salvar a transação no Tangle');
143   }
144
145   return info;
146 }
```

Fonte: Elaborado pelo Autor.

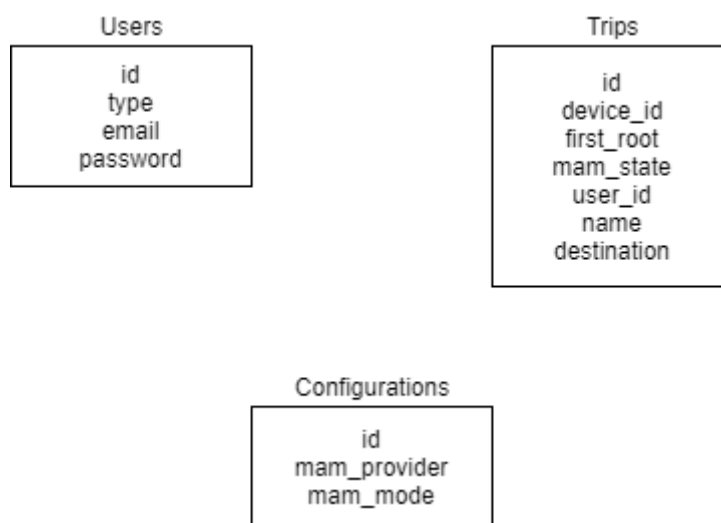
Cada viagem, dentro do banco MongoDB, é relacionada com o identificador de uma placa. Então, quando chega uma nova mensagem, o *backend* verifica se as informações são de uma viagem existente ou não. Caso seja, ele puxa do Mongo o estado da IOTA e então adiciona as novas informações no mesmo canal e salva esse estado atualizado. Caso não seja de uma viagem existente, ele simplesmente envia ao Tangle direto e salva uma nova viagem, com o estado da IOTA junto.

Como mencionado anteriormente, o *backend* está conectado a um banco de dados chamado MongoDB via Mongoose. O Mongoose é uma biblioteca ODM (Object Data

Modeling, ou em português, Modelagem de Dados de Objeto) para fazer a interface entre Node JS e MongoDB. Com essa interface, fica muito mais fácil criar os modelos e validar os dados a serem inseridos no banco.

Para que o *backend* possa cumprir o seu papel de gerenciar os usuários e enviar os dados ao Tangle, foi necessário criar três coleções dentro do MongoDB. A primeira: a coleção de Configurações (Configurations) não é obrigatória. Ela serve apenas para armazenar duas variáveis do Tangle, quando for trocado o modo (“mam_mode”) e trocar o provedor (“mam_provider”) ou para ambiente de testes ou de produção. A Figura 36 apresenta as coleções do banco.

Figura 36 - Coleções do banco de dados



Fonte: Elaborado pelo Autor.

A segunda tabela de usuários serve para guardar as informações da conta das pessoas que utilizam o sistema. Fora os campos básicos, como o *name*, email e *password*, tem um campo a mais que é o *type*, ou tipo em português. Os usuários podem ser de dois tipos: os “Distribuidores” ou “Consumidores”, que diferem um pouco as páginas que a conta tem acesso no *frontend*.

A terceira e última coleção armazena as informações necessárias das viagens. Percebe-se, primeiramente, dois campos: o “device_id” e o “mam_state”, com essas duas informações, é possível vincular os sistemas embarcados com o estado da IOTA (que contém o canal, a segurança, a próxima *root* e outros campos vistos nos capítulos anteriores). É com esses dois campos que o *backend* verifica se o pacote que chega é de uma viagem existente ou não. Primeiro, o *backend* verifica se já existe algum JSON dentro dessa coleção que tem o campo “device_id” igual ao “deviceUUID” que está chegando; se tiver, ele pega o estado relacionado com o valor que chegou e envia ao Tangle com as informações dentro de “mam_state”. Caso não exista um JSON com o campo “device_id” igual ao que está sendo recebido, é porque é uma nova viagem. Na primeira vez que o JSON é criado, o *backend* também salva o campo “first_root” que é utilizado mais tarde pelo *frontend* para pegar todas as mensagens enviadas ao canal. Por fim, há os campos “user_id”, “name” e “destination”, em que o primeiro representa de qual usuário é aquela viagem, o segundo é só um campo para dar

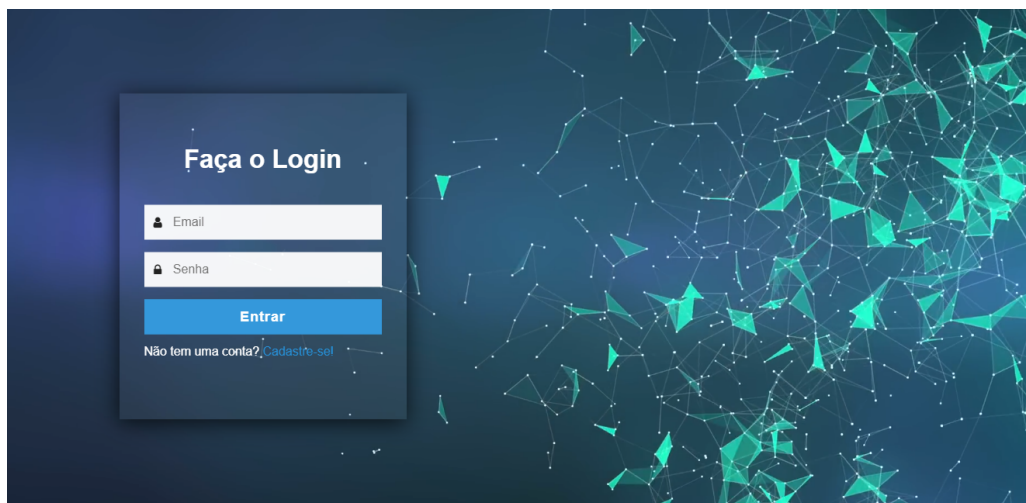
um nome à viagem e o terceiro se refere ao trajeto. Esses dois últimos campos são mais utilizados pelo *frontend*.

O *frontend* é escrito em React e totalmente responsivo. Fazendo com que usuários possam utilizar o sistema a partir de qualquer dispositivo, seja um celular, tablet, desktop entre outros. A única condição é que o dispositivo tenha um *browser*, ou outro interpretador de HTML qualquer.

Para que ambos serviços possam ser acessados pela nuvem da Ayga e por usuários, é necessário que ambos tenham um IP público. Para hospedar as aplicações, foi utilizado o Heroku (<https://www.heroku.com/>) que faz esse serviço de forma gratuita. Assim, a nuvem da Ayga, por meio do Callback mencionado anteriormente, consegue enviar os dados recebidos e usuários conseguem acessar o *front-end* em qualquer lugar, desde que tenha uma conexão com a internet.

Quando o sistema é acessado, o usuário verá a tela da Figura 37, que é a tela inicial do sistema. A partir dela é possível fazer *login* ou registrar uma nova conta. Na hora do registro, o usuário poderá escolher entre dois tipos de contas: a de distribuidor e a de consumidor. Com o primeiro tipo de conta, o usuário poderá visualizar viagens e cadastrar viagens de sua empresa. Enquanto que a conta de consumidor tem acesso apenas a viagens já cadastradas por distribuidores.

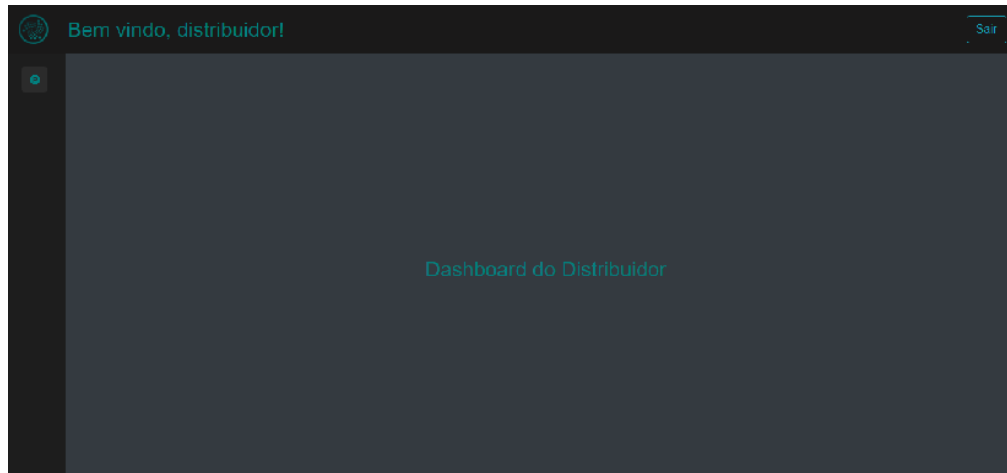
Figura 37 - Tela inicial do sistema



Fonte: Elaborado pelo Autor.

Como o foco do presente trabalho é que o resultado final seja uma prova de conceito, então o *dashboard* apresentado na Figura 37 acaba tendo poucas informações. Ele apresenta o nome do usuário logado e, na esquerda, as suas opções disponíveis de navegação. O consumidor pode ver as viagens já cadastradas dentro do sistema, enquanto que o distribuidor tem uma opção a mais, a de cadastrar a viagem. A Figura 38 ilustra a tela inicial depois de logado.

Figura 38 - Dashboard



Fonte: Elaborado pelo Autor.

A tela de cadastro de viagem é bem simples, Figura 39. A página consiste apenas de um formulário de três campos. O primeiro é para a inserção do ID da placa, o segundo é dado um nome a viagem (normalmente de um produto) e o terceiro é o trajeto que o produto fará. Então, por exemplo, o distribuidor acessa à página, insere o ID do sistema embarcado que está dentro de um caminhão e coloca o nome como “Batata do Lote 1000”, assim, ficará disponível para os usuários do sistema que aquela viagem está coletando informações de um lote que está transportando batatas. Caso o distribuidor insira um identificador da placa que não existe, o sistema retornará como um erro.

Figura 39 - Cadastro de viagens

A screenshot of a web application form titled 'CADASTRAR UMA VIAGEM'. The form is centered on a dark background. It contains three input fields: 'ID da placa', 'Nome do Produto', and 'Trajeto'. Below these fields is a teal 'Adicionar' button. The top navigation bar is dark with a circular profile icon on the left, the text 'Bem vindo, Daniel Transportes LTDA!' in the center, and a 'Sair' button on the right.

Fonte: Elaborado pelo Autor.

Ao acessar à página de visualização de alguma viagem, é possível ver de forma gráfica as duas variáveis obtidas. A Figura 40 ilustra como o *front-end* obtém os dados do *Tangle*.

Figura 40 - Leitura do dado do *Tangle*

```
Mam.init(config.mam_provider);

async function addListenerMam() {
  const result = await Mam.fetch(product.first_root, config.mam_mode);

  let msgs = [];

  if (result && result.messages) {
    result.messages.forEach(message => {
      const msg = JSON.parse(trytesToAscii(message));
      msgs.push(msg);
    });

    setMessage(Object.assign({}, msgs));
  }
}

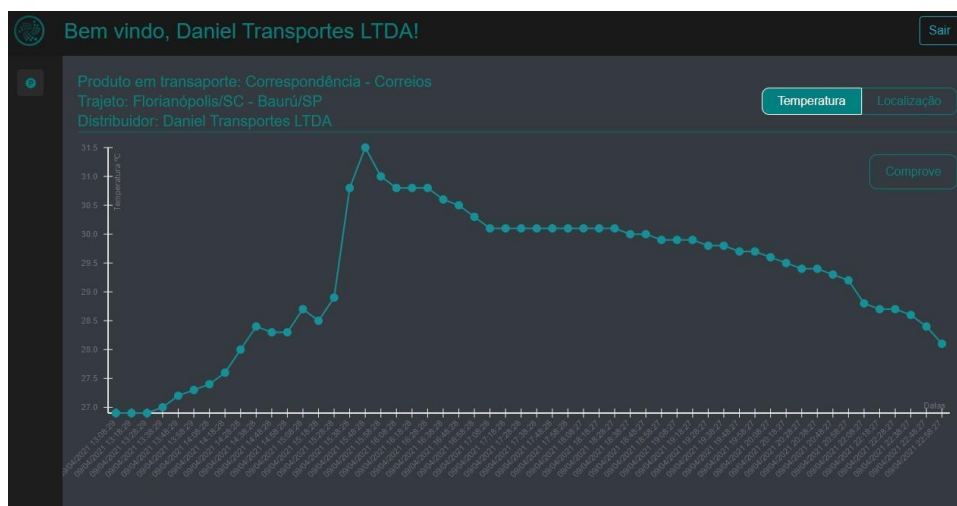
addListenerMam();
interval = setInterval(addListenerMam, 30000);
```

Fonte: Elaborado pelo Autor.

Primeiro, é inicializada uma nova instância do *Mam*, utilizando o *Mam.init()*, depois é utilizado o método *fetch()*, passa o primeiro endereço do canal ("*first_root*") para pegar todas as mensagens do canal de forma síncrona. Então, são convertidas todas as mensagens de *trytes* para *bytes* e salva as mensagens, para que sejam utilizadas pelo *front-end*.

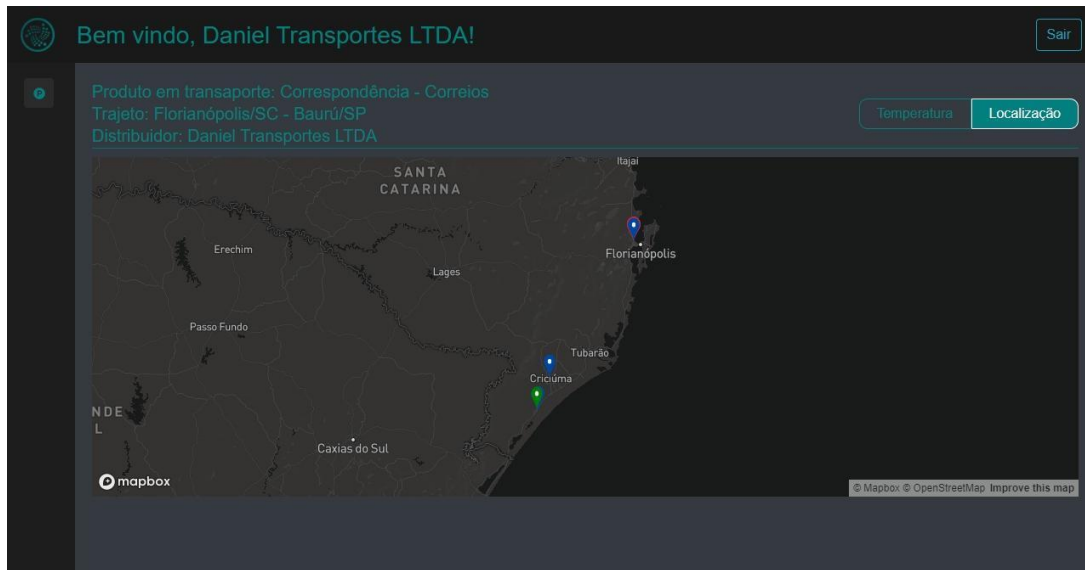
As Figuras 41 e 42 apresentam a visualização de temperatura e da localização durante o tempo. A primeira variável obtida, a temperatura, é mostrada em um gráfico simples, em que o eixo x representa o tempo (com data e hora) e o eixo y o valor da temperatura em si na data ilustrada. Já a localização é representada por pontos no mapa. O primeiro ponto recebe um marcador vermelho, os intermediários são os marcadores azuis e o último marcador é representado com a cor verde.

Figura 41 - Temperatura da entrega



Fonte: Elaborado pelo Autor.

Figura 42 - Localização da entrega



Fonte: Elaborado pelo Autor.

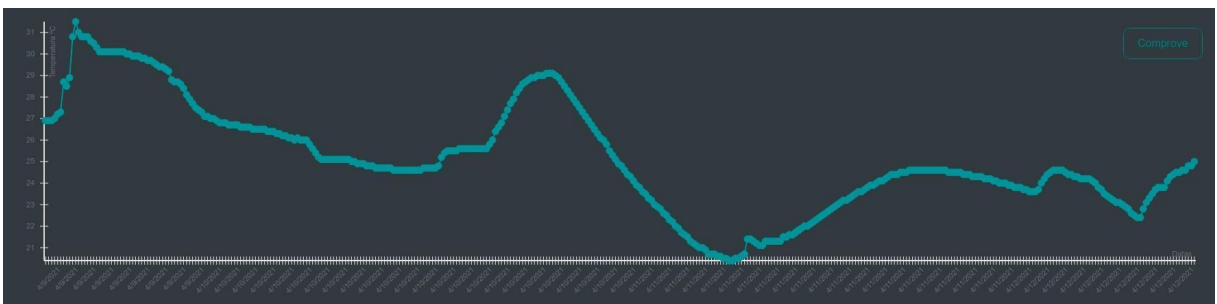
Vale comentar os dois pontos na página de visualização de produtos. O primeiro é o porquê do mapa ser apenas pontos e não um trajeto inteiro, isso se deve, pois o sistema só consegue garantir os pontos em que a variável foi lida e não o intermédio entre esses pontos. A segunda questão que vale ser ressaltada é que o *frontend* precisa de duas etapas para conseguir esses dados. Primeiro é enviada uma requisição do tipo GET ao *backend* para pegar uma variável que está armazenada no banco, a “first_root”. Com essa variável obtida, o *frontend* consegue criar um *listener* (ou ouvinte) do canal das mensagens no Tangle da IOTA. Logo, o *frontend*, a partir disso, obtém os valores guardados no Tangle e disponibiliza na tela do usuário no momento exato. Se chegar uma mensagem nova no Tangle, ela é disponibilizada de forma automática ao usuário, sem precisar de iteração dele, ou seja, de forma automática, os novos valores são mostrados ao vivo para o consumidor/distribuidor.

7. TESTES E VALIDAÇÕES

A partir da prova de conceito desenvolvida e da integração de todos os componentes do sistema, foram realizados testes e validações com o objetivo de avaliar seu funcionamento e sua viabilidade de aplicação em um contexto real. O experimento teve dois principais focos: verificar a qualidade da rastreabilidade de produtos em movimento entre cidades, verificando a qualidade da transmissão dos dados em movimento; e a avaliação da aplicação Web proposta como meio de transparência dos dados aos consumidores e distribuidores, que estejam interessados no contexto do produto.

O experimento consistiu em fazer com que o *hardware* fosse movido entre cidades e, conseqüentemente, poder acompanhar o seu movimento e a temperatura do ambiente durante todo o percurso. Para realizar esse teste, o sistema embarcado foi enviado pelos correios a partir da cidade de Araranguá até a cidade de Bauru, em São Paulo. As Figuras 43 e 44 apresentam a temperatura e o trajeto, respectivamente.

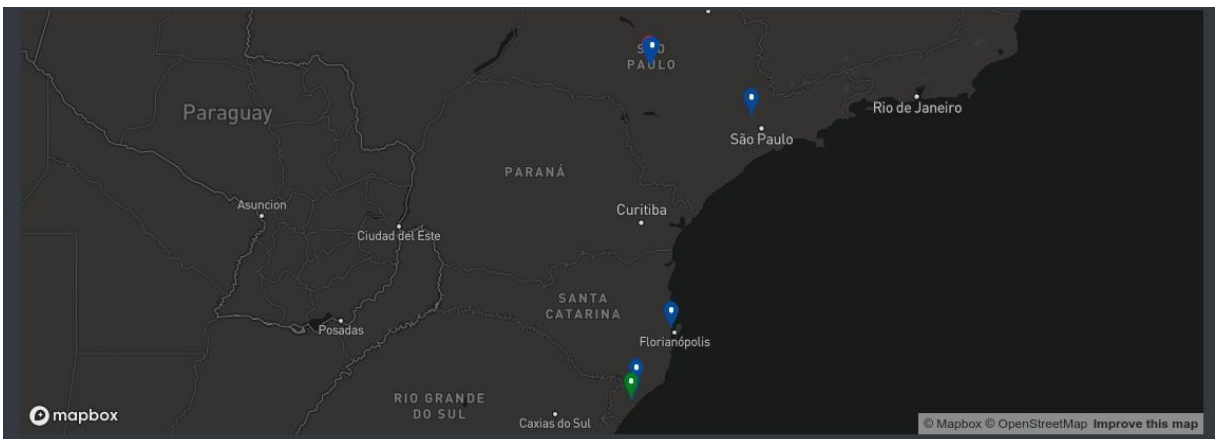
Figura 43 - Dados de temperatura



Fonte: Elaborado pelo Autor.

A Figura 44 tem cinco principais regiões com pontos registrados, em verde e primeiro, Araranguá, seguido de Criciúma e Florianópolis em Santa Catarina. Depois, em São Paulo, Vila Nova e Baurú (em azul/vermelho) e mais acima do que todos os outros pontos.

Figura 44 - Dados de localização



Fonte: Elaborado pelo Autor.

Para ajudar a visualização neste trabalho, foram construídos os Apêndices A e B, que contêm os dados com as datas convertidas (de *timestamps* para uma data mais fácil de visualização e fuso horário brasileiro). Uma das observações que pode ser feita é em relação à temperatura, que atinge o seu pico em torno das 17 horas, quando a entrega está passando pela cidade de Criciúma. Depois deste pico, a temperatura vai baixando até estabilizar por volta dos 26°C, em Florianópolis.

Tanto para a localização quanto para a temperatura, o sistema foi configurado para tentar receber o valor a cada 10 minutos. Ou seja, nas tabelas quando tiver um “espaço” maior do que esse tempo, quer dizer que o dado foi perdido. A temperatura teve poucas perdas de dados, mas a localização já teve mais dados que não chegaram ao *Tangle*. Isso faz sentido, pois, como a localização é feita a partir da triangularização de roteadores Wi-Fi, é esperado que em lugares como a BR-101 tenha muito menos possibilidade de conseguir tais valores.

Vale ressaltar que, mesmo com as perdas de alguns dados, os resultados ainda são satisfatórios. Levando em consideração que a rede SigFox tem cobertura em Araranguá, Criciúma e depois apenas em Florianópolis e todo o trajeto é “obscuro”, foi possível obter a localização nessas três cidades que têm sinal do trajeto. Em relação à temperatura, o *hardware*, mesmo sem sinal da rede, continuou monitorando o ambiente e, quando teve sinal, enviou todos eles, fazendo com que o gráfico de temperatura “fluísse” de forma convincente. Mas, no trajeto Florianópolis até São Paulo, existe uma área muito grande de cobertura por várias cidades do trajeto.

Um dos possíveis motivos para não ter conseguido dados de localização nesse trajeto é que talvez a encomenda não tenha passado por cidades como Joinville e Curitiba, que tem cobertura. Outro motivo, mais provável, é que pela localização ser obtida por Wifi faz sentido que não tenha sido possível obter tais dados enquanto o *hardware* estava em movimento. Pelo mapa e pela Tabela 7, percebe-se que isso de fato aconteceu e foi possível apenas obter a localização quando o sistema embarcado ficou um bom tempo em uma certa localização. Essa perda é de se esperar, se for pensado que um caminhão tem velocidade média de 80km/h nas rodovias, talvez ele passe de forma muito rápida para que possam ser escaneados os MACs de roteadores nas redondezas. Sem falar que áreas rurais têm muito menos conexões Wi-Fi quando comparadas com centros urbanos.

Neste teste, não foi enviado nenhum produto crítico, como por exemplo comida (frutas, arroz, etc). Então, só é possível ver que os dados de temperatura foram obtidos quase que por inteiro, mas vale ressaltar que a temperatura de 31°C obtida em Criciúma poderia ter estragado certas mercadorias na viagem.

Também é possível notar outro padrão nos dados de localização, que é a quarta coluna do Apêndice A, o raio da margem de erro. Quanto mais fraco é um sinal do roteador, maior é esse raio. Esse valor é mais alto quando está no Centro de Logística dos Correios de Vila Nova, pois esse prédio fica afastado do centro urbano da cidade e há menos sinais de wi-fi ao redor.

Exceto esse pico de temperatura em Criciúma, todo o resto da viagem ocorreu conforme o esperado. Foram quatro dias de Araranguá até Baurú, que é um tempo bom para essa viagem, ainda mais se considerar que o sistema embarcado foi enviado em uma sexta-feira fim de tarde. De mais, as temperaturas ficaram em um clima agradável.

Este mesmo teste também avalia a aplicação Web desenvolvida. Foi necessário observar principalmente dois pontos: primeiro, a sua capacidade de receber os dados do *hardware* e enviá-los ao *Tangle*, e segundo como o sistema consegue gerar uma boa experiência de usuário quando dá transparência aos dados.

Foram poucos dados que foram perdidos por queda do servidor, praticamente todos os dados que chegaram ao *back-end*, conseguiram chegar ao *Tangle*. Já em relação ao *front-end*, foi notada uma demora para mostrar os dados na tela do usuário. Isso foi causado pela escolha de trazer os dados diretamente do *Tangle*. Pegar todos os dados necessários diretamente do *Tangle* pode demorar. Mesmo com a demora, ainda é um tempo considerado rápido, se levar em consideração todo o trâmite dos pacotes enviados a partir da placa. Fora isso, todo o restante foi realizado com sucesso, logo é possível comprovar os dados, não é possível alterar qualquer coisa e é possível acompanhar quase em tempo real, conforme esperado.

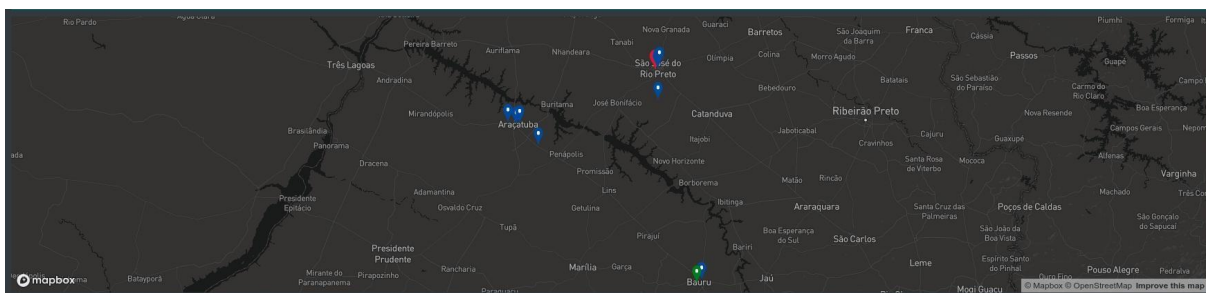
Também foi realizado um segundo teste, em que uma pessoa andou com o *hardware* dentro de um caminhão, em cima do painel, no banco do motorista. Durante a primeira manhã, quando foram percorridas áreas com sinal da rede SigFox, foram perdidos poucos pacotes. As Figuras 45 e 46 apresentam a temperatura e a localização respectivamente.

Figura 45 - Gráfico de temperatura do segundo teste



Fonte: Elaborado pelo Autor.

Figura 46 - Localização do segundo teste



Fonte: Elaborado pelo Autor.

Para auxiliar na visualização destes gráficos, foram construídos os Apêndices C e D, que contêm as informações guardadas dentro do *Tangle* do segundo teste. A região em que o caminhão percorreu o segundo teste é uma região urbana e pode haver mais redes Wi-Fi. Outra questão é a velocidade, que pode ter influenciado na obtenção de sinais de localização. Uma terceira possibilidade é o lugar em que estava o *hardware*, pois nos correios é possível

que o sistema embarcado estivesse no meio a várias caixas, enquanto que, no segundo teste, ele estava em um lugar de melhor acesso.

8. CONSIDERAÇÕES FINAIS

Neste trabalho foi proposto uma prova de conceito para o rastreamento de produtos em transporte que adota uma abordagem de um dispositivo com sensores com uma conectividade com a rede SigFox, tem como banco de dados para os sensores o *Tangle* da IOTA e busca promover segurança e imutabilidade dos dados. Também tem uma interface Web para monitoramento ao vivo dos dados coletados do produto.

Baseado, neste modelo, foi desenvolvido um protótipo inicial para a realização dos testes e validações que consistiu em um sistema embarcado que monitora a localização e temperatura; também o desenvolvimento de um sistema Web que armazena as informações no *Tangle* e busca os dados dinâmicos do mesmo local, mostrando ao usuário quase em tempo real os eventos.

Foram realizados dois testes com a prova de conceito para avaliar a conectividade da tecnologia SigFox com um produto em movimento e, também, em relação ao *Tangle* da IOTA em relação à segurança e à experiência de usuário utilizando essa tecnologia como banco de dados de uma aplicação. Nessa etapa foi analisada a área de cobertura da rede e a capacidade de recebimento de sinais em movimento, bem como a velocidade, segurança e experiência de usuário quando em conexão ao *Tangle*.

A partir dos experimentos, foi possível validar e perceber a importância da utilização da rede SigFox como meio de comunicação para este caso. Visto que ela consome pouca energia, consegue ter uma boa quantidade de mensagens entregues mesmo quando em movimento, longo alcance e uma boa cobertura. Além disso, a integração com os serviços em nuvem (SigFox Cloud, Ayga e Here) em combinação com as tecnologias utilizadas para desenvolvimento da aplicação Web se mostraram eficientes para monitoramento de forma simples e rápida de visualização de todos os dados obtidos.

Apesar de ter resultados satisfatórios, adotar essa arquitetura em um cenário real necessita de alterações e pesquisas que estão fora do escopo deste trabalho, por meio do qual foi realizada apenas uma análise inicial a partir de uma prova de conceito.

O primeiro fator que deve ser considerado para implementação deste modelo é a cobertura da rede SigFox. Em um local como Santa Catarina, este trabalho obteve uma boa quantidade de sinal para fazer leituras, mas em outros estados a cobertura pode não ser tão abrangente assim. Além disso, faz-se necessário um projeto de um sistema embarcado considerando melhor os seus componentes, principalmente em relação ao tamanho da mensagem e também ao consumo da bateria.

O sistema Web também precisa ter um estudo para obter um melhor resultado em relação à experiência de usuário, para que não tenha uma demora considerável para carregar e mostrar os dados a quem esteja interessado.

8.1 Trabalhos Futuros

A partir do que foi elaborado neste trabalho, sugere-se como trabalhos futuros para o desenvolvimento do modelo proposto os itens colocados a seguir:

1. Utilizar como meio de comunicação a rede de celulares 5G, dependendo de como estiver a sua cobertura no momento do estudo.

2. Outra alternativa para a comunicação seria a LoRaWAN, onde pode-se comprar o *hardware*, retirando nuvens de terceiros da solução.
3. Pensar em outra solução distribuída, pois o *Tangle* faz snapshots de tempo em tempo, perdendo o histórico.
4. Projetar e desenvolver um sistema embarcado de monitoramento para produtos em movimento, considerando o contexto de áreas não urbanas, especialmente a antena.
5. No sistema Web, guardar os dados obtidos em um banco de dados e também no *Tangle*, junto com uma *hash* dos dados, utilizando assim o *Tangle* apenas para conferência se os dados estão corretos e não carregar eles diretamente de lá.

REFERÊNCIAS

- AL-KASHOASH, H.; KEMP, A. Comparison of 6LoWPAN and LPWAN for the Internet of Things. **Australian Journal of Electrical and Electronics Engineering**. v. 13, n. 4, p. 268-274. 2017. Disponível em: https://www.researchgate.net/publication/316236998_Comparison_of_6LoWPAN_and_LPWAN_for_the_Internet_of_Things. Acesso em 30 abril 2020. DOI: 10.1080/1448837X.2017.1409920.
- ALFIAN, G. *et al.* Integration of RFID, Wireless Sensor Networks, and Data Mining in an e-Pedigree Food Traceability System. **Journal of Food Engineering**. v. 212, p. 65-75, nov. 2017. Disponível em: https://www.sciencedirect.com/science/article/pii/S0260877417302066?casa_token=6mUiyFd-f-8AAAAA:yovkhsX2fNsai9PLxnHLZndoJcJ3XciijEuWPSQvXKeIRDGn6CFnMS2maYry8XeDj7rEMX1KqTg. Acesso em 04 abril 2020. DOI: 10.1016/j.jfoodeng.2017.05.008.
- AUNG, M. M.; CHANG, Y. S. Traceability in a food supply chain: Safety and quality perspectives. **Food control**, v. 39, p. 172-184, 2014. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0956713513005811>. Acesso em: 04 abril 2020. DOI: 10.1016/j.foodcont.2013.11.007.
- AYGA. **WS2**. 2021. Disponível em: <https://www.ayga.com.br/produtos>. Acesso em: 02 maio 2021.
- BAKARE, B.; EKANEM, I., ALLEN, I. Appraisal of Global System for Mobile Communication (GSM) In Nigeria. **American Journal of Engineering Research (AJER)**. v. 6, p. 97-102, 2017. Disponível em: https://d1wqtxts1xzle7.cloudfront.net/53764234/N060697102.pdf?1499239118=&response-content-disposition=inline%3B+filename%3DAppraisal_of_Global_System_for_Mobile_Communication%28GSM%29_in_Nigeria.pdf&Expires=1619963984&Signature=EIxcYNWUX8W~UeOX87~6OZdIMRyABCR1isVdHX9CLO2EILdWFzSRbH8kRAv2yTeCDSluqUzSUZwqTXs2x6Fxiy0BLLxY81PfyEL8x3jFwIPG~UD75TKm~PFA-sFjqeJ2uVknR7v4PkPgj68yXQYZXLcoDFvbOJ7fNyVx7NCDE1vYj6HL6b9cBLL9t5b7QmG1yc1VIMP4TR1PteEtYA3yb2KivHLG-yCuEjlcU1f7JafNSIBk7~DtQn0GoAfynddGW~uLn9-ZKUHwDxz08cW5d4biX4Y0B1LRYjuO2pWnOhvprsmGcGMY5WjQjc~8d6AW6vvWtyu4o64qv1vqT60Alw__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA. Acesso em: 04 abril 2020.
- BENCIC, F., M.; ZARKO, I. P. Distributed Ledger Technology: Blockchain Compared to Directed Acyclic Graph. **Cornell University p. 1-6**, 2018. Disponível em: <https://arxiv.org/pdf/1804.10013.pdf>. Acesso em 20 maio 2020.
- BIRADAR, N. B.; DYVIA, M. IOTA-Next Generation Blockchain. **International Journal Of Engineering And Computer Science**. v., n.4, p. 23823-23826, 2018. Disponível em: <http://ijecs.in/index.php/ijecs/article/view/4007>. Acesso em 25 maio 2020. DOI: 10.18535 / ijecs / v7i4.05.
- BU, G., HANA, W., POTOP-BUTUCARU, M.. Metamorphic IOTA. **Cornell University**. p. 1-8, **jul** 2019. Disponível em: <https://arxiv.org/pdf/1907.03628.pdf>. Acesso em 25 maio 2020.

BUCHMAN, E.. **Tendermint: Byzantine Fault Tolerance in the Age of Blockchains**. 2016. Dissertação (Mestrado em Engenharia) The University of Guelph. Guelph. Ontario, Canada. 2016. Disponível em: atrium.lib.uoguelph.ca/xmlui/bitstream/handle/10214/9769/Buchman_Ethan_201606_MAsc.pdf?sequence=7&isAllowed=y. Acesso em 20 maio 2020.

CHEN, T. *et al.* Batch-based traceability for pork: A mobile solution with 2D barcode technology. **Food Control**. v. 107, jan. 2020. Disponível em https://www.sciencedirect.com/science/article/pii/S0956713519303597?casa_token=aOt9unAfwq0AAAAA:ZvQ-vAD9nBT_WUw5rHo9QLXkjHMaGWU1iI4NsNnRWz3z4DYkf6eAXIfgkTIo30voQMjF1W4Fnbs. Acesso em 30 abril 2020.

COULOURIS, G. DOLLIMORE, J. KINDBERG, T. **Sistemas Distribuídos: Conceitos e Projetos**. 4ª edição. 2007. Disponível em: <http://www.inf.ufsc.br/~bosco.sobral/ensino/ine5645/coulouris.pdf> Acesso em: 15/05/2021.

DABBENE, F., GAY P., TORTIA C. Traceability Issues In Food Supply Chain Management: A Review. **Biosystems engineering**, v. 120, p. 65-80, abril 2014. Acesso em: 05 abril 2020. DOI: 10.1016/j.biosystemseng.2013.09.006.

DIVYA, M., BIRADAR, N. IOTA-Next Generation Blockchain. **International Journal Of Engineering And Computer Science**, v.7, n. abril 2018. Disponível em: https://www.researchgate.net/publication/324232469_IOTA-Next_Generation_Block_chain#:~:text=IOTA%20is%20a%20revolutionary%20new,Chain%20and%20also%20no%20Miners. Acesso em 05 abril 2020. DOI: 10.18535/ijecs/v7i4.05.

EUGSTER, P. T., *et al.* The Many Faces of Publish/Subscribe. **ACM Computing Surveys**, v. 35, n. 2, pp. 114–13, , Jun 2003. Disponível em: <https://infoscience.epfl.ch/record/165428/files/10.1.1.10.1076.pdf>. Acesso em: 17 maio 2020.

FAN, B. *et al.* **Improving continuous traceability of food stuff by using barcode-RFID bidirectional transformation equipment: Two field experiments**. v. 98, p. 449-456, abril 2019. DOI: <https://doi.org/10.1016/j.foodcont.2018.12.002>. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0956713518306029>. Acesso em: 25 abril 2020.

FENECH, G. IOTA - Fulfilling the Promise of Blockchain. **Revista Forbes**, nov. 2018. Disponível em: <https://www.forbes.com/sites/geraldfenech/2018/11/20/iota-fulfilling-the-promise-of-blockchain/#6df75c7d5735> Acesso em: 05 abril 2020.

GALVEZ, J.F., MEJUTO, J., SIMAL-GANDARA, J. Future challenges on the use of blockchain for food traceability analysis. **TrAC Trends in Analytical Chemistry**, v.107, p. 222-232, out. 2018. Disponível em: https://www.sciencedirect.com/science/article/pii/S0165993618301304?casa_token=W6B7F49s0W4AAAAA:ypPItGGX6IVLu7YdgjowaeJZExa-tD52aBFw1hhLH44qqj1lT-adYIoGq8T-OLVCNJ7Fj02ldJk. Acesso em: 05 abril 2020. DOI: 10.1016/j.trac.2018.08.011.

GOMEZ, C. *et al.* A Sigfox Energy Consumption Model. **Sensors**, v. 19, n. 3, fev. 2019. Disponível em; <https://www.mdpi.com/1424-8220/19/3/681> Acesso em 05 maio 2020. DOI: 10.3390/s19030681.

HARRIS, A. L., REA, A.. **Web 2.0 and Virtual World Technologies: A Growing Impact on IS Education**. Disponível em: <aisel.aisnet.org/cgi/viewcontent.cgi?article=1271&context=jise> Acesso em 23 jun 2020.

HE, R. *et. al.*. High-Speed Railway Communications: From GSM-R to LTE-R. **IEEE Vehicular Technology Magazine**. v. 11, n.3, p. 49-58, set. 2016. Disponível em: https://ieeexplore.ieee.org/abstract/document/7553613?casa_token=-n_ioRbfHq0AAAAA:t6XJlxVAWHK6ubO35a8LE1-eIKRf3PWtYacRHHtf4X7IYwx5rJXSfm79h7xUTKBVtNCGXN3MUnI. Acesso em: 10 maio 2020. DOI: 10.1109/MVT.2016.2564446.

HU, J., *et. al.* Modeling and implementation of the vegetable supply chain traceability system. **Food Control**. v. 30, p. 341–353, mar. 2013. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0956713512003696>. Acesso em: 14 abril 2020. DOI: 10.1016/j.foodcont.2012.06.037.

HUANG, P., PSOUNIS, K. Optimal backhauling for dense small-cell deployments using mmWave links. **Computer Communications**. 138, 32-44, 2019. DOI: 10.1016/j.comcom.2019.03.002. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0140366418307849> Acesso em 29 abril 2020.

JIMÉNEZ, L. M., PUERTO, R., PAYÁ, L.. **Sistemas Distribuídos: Arctectura y Aplicaciones**. Elche: Universidad Miguel Hernández. 2017. Disponível em: https://books.google.com.br/books?hl=pt-BR&lr=&id=2W41DwAAQBAJ&oi=fnd&pg=PR9&dq=+Sistemas+Distribu%C3%ADdos:+Arctectura+y+Aplicaciones&ots=9kBtixIoWk&sig=mhzxKv2YKBjL7i0qTMoaUyzqEXo&redir_esc=y#v=onepage&q=Sistemas%20Distri bu%C3%ADdos%3A%20Arctectura%20y%20Aplicaciones&f=false Acesso em: 20 maio 2020

LEAL, R. L. V. **Blockchain e Iinternet das coisas: Aplicações e Iniciativas** . 2017. Disponível em: https://www.cpqd.com.br/wp-content/uploads/2017/11/Whitepaper_Blockchain_e_IoT_-_apli cac%CC%A7o%CC%83es_e_iniciativas_v2.pdf. Acesso em: 20 maio 2020.

LEI W. *et al.* From 4G to 5G: Use Cases and Requirements. **5G System Design**. v. 1, n.33, jan. 2020.. Disponível em: https://www.researchgate.net/publication/335725483_From_4G_to_5G_Use_Cases_and_Req uirements. Acesso em: 28 abril 2020. DOI: https://doi.org/10.1007/978-3-030-22236-9_1

LAN CZECH IOT CLOUD. **LPWAN SIGFOX Node Datasheet**. Dispoível em: https://storage.sbg.cloud.ovh.net/v1/AUTH_669d7dfced0b44518cb186841d7cbd75/prod_med ias/j1yudrck_lpwan-sigfox-node-datasheet-v1.pdf. Acesso em: 28 jun. 2020.

MEKKI, K., *et. al.* A comparative study of LPWAN technologies for large-scale IoT deployment. **ICT Express**. v. 5, n. 1, p. 1-7, mar. 2019. Disponível em:

<https://www.sciencedirect.com/science/article/pii/S2405959517302953>. Acessado em 05 abril 2020.

METZNER, V. C. FRAY, R.S.; CUGNASCA, C. E. Modelo de Rastreabilidade de Medicamentos Usando RFID e o Conceito de Internet das Coisas. **Anais Anpet**, Curitiba, out. 2015. Disponível em: http://www.anpet.org.br/ssat/interface/content/autor/trabalhos/publicacao/2014/267_AC.pdf. Acesso em: 27 abril 2020.

MIKHAYLOV, K. PETÄJÄJÄRV, J. HÄNNINEN, T. Analysis of the Capacity and Scalability of the LoRa Wide Area Network Technology. **Conference: European Wireless**. Finlândia. maio 2016. Disponível em: https://www.researchgate.net/publication/303917895_Analysis_of_the_Capacity_and_Scalability_of_the_LoRa_Wide_Area_Network_Technology. Acesso em: 04 maio 2020.

MISHRA, D. K., *et al.* Traceability as an integral part of supply chain logistics management: an analytical review. **Cornell University**, nov. 2018. Disponível em: <https://arxiv.org/abs/1811.06358> Acesso em: 04 abril 2020.

MONTEIRO, E. J.; RIVEIRO, L.; BARRETO, R. Uma Ferramenta de Suporte ao Ensino de Modelagem de Sistemas Distribuídos Críticos: Uma Experiência Prática. **Semantic Scholar**, 2018. Disponível em: <https://pdfs.semanticscholar.org/92c0/f4c1548b9a446b4a7e662c05df7b5a31f9e9.pdf>. Acesso em: 16 maio 2020. DOI: <http://dx.doi.org/10.5753/cbie.sbie.2018.218>

NGUYEN, H. V.. **End-to-End E-commerce web application, a modern approach using MERN stack**.Helsinque: Metropolia University of Applied Sciences, 2020. Disponível em: https://www.theseus.fi/bitstream/handle/10024/339760/Nguyen_Hung.pdf?sequence=2&isAllowed=y Acessado em 28 jun. 2020.

NIU, Y. *et al.* A survey of millimeter wave communications (mmWave) for 5G: opportunities and challenges. **Wireless Netw.** v. 21, p. 2657–2676, abril 2015. Disponível em: <https://link.springer.com/article/10.1007/s11276-015-0942-z>. Acesso em: 29 abril 2020. DOI: <https://doi.org/10.1007/s11276-015-0942-z>.

PARIKH, J.; BASU, A. Technologies Assisting the Paradigm Shift from 4G to 5G. **Wireless Pers Communications**, v.112, p. 481-502, . 2020. Disponível em: <https://link.springer.com/content/pdf/10.1007/s11277-020-07053-3.pdf>. Acesso em: 28 abril 2020. DOI: <https://doi.org/10.1007/s11277-020-07053-3>.

PECK, M. Blockchain world - Do you need a blockchain? This chart will tell you if the technology can solve your problem. **IEEE Spectrum**. v, 54, n.10, p. 38-60, set. 2017. Disponível em: <https://ieeexplore.ieee.org/abstract/document/8048838>. Acesso em; 05 abril 2020. DOI: 10.1109/MSPEC.2017.8048838.

PERLMAN, R.. **Blockchain: Hype or Hope?, Summer**, v. 42, n.2, . 2017 Disponível : https://www.usenix.org/system/files/login/articles/login_summer17_13_perlman.pdf. Acesso em: 24 maio 2020.

PINJALA, S., SIVALINGAM, K. DCACI: A Decentralized Lightweight Capability Based Access Control Framework using IOTA for Internet of Things. 2019 IEEE 5th World Forum on Internet of Things (WF-IoT). **Quinto Fórum Mundial IEEE**, p. 13-18, 2019. Disponível em: <https://ieeexplore.ieee.org/abstract/document/8767356>. Acesso em: 24 maio 2020. DOI: 10.1109/WF-IoT.2019.8767356.

POPOV, S. The Tangle. **abril** 2016. Disponível em: https://iotatoken.com/IOTA_Whitepaper.pdf. Acesso em: 25 maio 2020.

PORTER, P.; YANG, P. P. S.; XI, X.. The Design and Implementation of a RESTful IoT Service Using the MERN Stack. **2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems Workshops (MASSW)**, p 140-145, 2019. Disponível em: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9059381>. Acesso em: 23 jun. 2020. DOI: 10.1109/MASSW.2019.00035.

QUERINI, M. *et al.* 2D Color Barcodes for Mobile Phones. **International Journal of Computer Science and Applications**. v. 8, n. 1, p. 136-155, 2011. Disponível em: <https://core.ac.uk/download/pdf/53848928.pdf>. Acesso em: 27 abril 2020.

RAY, P. P. A Survey on Internet of Things Architectures. **Journal of King Saud University – Computer and Informations Sciences**. v. 30, n.3, p. 291-319, 2018. Disponível em: <https://reader.elsevier.com/reader/sd/pii/S1319157816300799?token=4A6C6CB343B572EFF93B7A03388E0FF525C012823341CC0B921A2B5049177DF7580DF05D63906EBCBB253E586A32D548&originRegion=us-east-1&originCreation=20210502012859>. Acesso em: 30 abril 2020.

REGATTIERI, A.; GAMBERI, M.; MANZINI, R. Traceability of Food Products: General Framework and Experimental Evidence. **Journal of Food Engineering**. v. 81,p. 347-356, jul. 2007. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0260877406006893>. Acesso em: 14 abril 2020. DOI: 10.1016/j.jfoodeng.2006.10.032.

ROHINI, C. G.; SATYANARAYANA, M. V.; SRIKANTAN, A.V. Analysis of Beam Division Multiple Access in Massive MIMO Communication Systems. **American Journal of Computer Science and Information Technology**, 2016. Disponível em: <https://www.imedpub.com/abstract/analysis-of-beam-division-multiple-access-in-massive-mimo-communication-systems-17184.html>. Acesso em: 29 abril 2020.

SEGURA V. *et al.* Towards industrial internet of things: Crankshaft monitoring, traceability and tracking using RFID. **Robotics and Computer-Integrated Manufacturing**. v. 41, p. 66-77, out. 2016. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0736584516300576>. Acesso em: 28 abril 2020. DOI: 10.1016/j.rcim.2016.02.004.

SIGFOX RADIO. **Radio Configurations**. França: SigFox, 2021. <https://build.sigfox.com/sigfox-radio-configurations-rc>. Acesso em: 31 jan. 2021.

SINHA, R.; YIQIAO, W.; HWANG, S. A survey on LPWA technology: LoRa and NB-IoT. **ICT Express**, v. 3, n. 1, p. 14-21, mar. 2017. Disponível em:

<https://www.sciencedirect.com/science/article/pii/S2405959517300061>. Acesso em: 11 maio 2020. DOI: 10.1016/j.ict.2017.03.004.

SIGFOX. **Sigfox Technical Overview**. 2017. Disponível em <https://www.disk91.com/wpcontent/uploads/2017/05/4967675830228422064.pdf>. Acesso em: 05 maio 2020.

STEEN, M. V.; TANENBAUM, A. S. **Distributed Systems**. 3. Ed. Pearson Education,. Thirdy Edition, 2013. TARJAN, L., *et. al.* A readability analysis for QR code application in a traceability system. **Computers and Electronics in Agriculture**. v.109, p. 1–11, nov. 2014. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0168169914002142>. Acesso em: 15 abril 2020. DOI: 10.1016/j.compag.2014.08.015.

THAKUR, M. *et al.* Traceability in the Bulk Grain Supply Chain. Resource Magazine. **Journal of Food Engineering**, v. 95, n.4, dez. 2009. Disponível em: <https://www.sciencedirect.com/science/article/abs/pii/S0260877409003264?via%3Dihub>. Acesso em: 13 abril 2020. DOI: <https://doi.org/10.1016/j.jfoodeng.2009.06.028>.

WUST, K.; GERVAIS, A. Do you need Blockchain?. **Crypto Valley Conference on Blockchain Technology (CVCBT)**, p, 45-54, jul. 2018. Disponível em: <https://eprint.iacr.org/2017/375.pdf>. Acesso em: 24 maio 2020. DOI: 10.1109 / CVCBT.2018.00011

YAGA, D. *et al.* Blockchain Technology Overview. **National Institute of Standards and Technology Internal Report 8202, out**. 2018. Disponível em: <https://arxiv.org/ftp/arxiv/papers/1906/1906.11078.pdf>. Acesso em: 18 maio 2020. DOI: e free <https://doi.org/10.6028/NIST.IR.8202>.

YOUSUF A. *et al.* **Barcode Detection of Video Images by Using MATLAB SIMULINK**. . Computer Science and Engineering, 2019, v. 9, n.1, p. 12-19, 2019. Disponível em: https://www.researchgate.net/profile/Md-Harun-Rashid-2/publication/337493086_Barcode_Detection_of_Video_Images_by_Using_MATLAB_SIMULINK/links/5ddbc9a692851c1fedafc828/Barcode-Detection-of-Video-Images-by-Using-MATLAB-SIMULINK.pdf. Acesso em: 25 abril 2020. DOI: 9. 12-19. 10.5923/j.computer.20190901.03.

APÊNDICES

APÊNDICE A - TABELA DE LOCALIZAÇÃO DO PRIMEIRO TESTE

Data e Hora	Latitude	Longitude	Raio (m)
09/04/2021 13:58:34	-28.941261	-49.49532448	66
09/04/2021 14:08:34	-28.9410586	-49.49549284	87
09/04/2021 14:18:34	-28.941261	-49.49532448	66
09/04/2021 14:48:34	-28.9412633	-49.49532388	67
09/04/2021 14:58:34	-28.9412606	-49.49533939	87
09/04/2021 15:08:34	-28.9412633	-49.49532388	68
09/04/2021 15:18:34	-28.9412633	-49.49532388	67
09/04/2021 15:28:34	-28.9411345	-49.4954028	68
09/04/2021 15:38:34	-28.945113	-49.49107777	82
09/04/2021 15:48:34	-28.945113	-49.49107777	82
09/04/2021 15:58:34	-28.9452866	-49.49132	96
09/04/2021 16:18:34	-28.9347376	-49.48477233	123
09/04/2021 16:28:34	-28.9346588	-49.48472322	168
09/04/2021 16:38:34	-28.9344288	-49.48356474	179
09/04/2021 16:48:34	-28.9344443	-49.48501898	78
09/04/2021 16:58:33	-28.9361735	-49.48298507	120
09/04/2021 17:38:33	-28.6822722	-49.38505736	58
09/04/2021 21:38:33	-27.576069	-48.61474422	202
09/04/2021 21:58:33	-27.5766501	-48.61414438	112
09/04/2021 22:08:33	-27.5777112	-48.61448093	121
09/04/2021 22:18:33	-27.5747941	-48.61361851	106
09/04/2021 22:28:33	-27.5747941	-48.61361851	106
09/04/2021 22:38:33	-27.576069	-48.61474422	202

09/04/2021 23:08:33	-27.576069	-48.61474422	202
10/04/2021 00:08:32	-27.5754331	-48.61391827	91
10/04/2021 00:28:32	-27.5754331	-48.61391827	91
10/04/2021 01:48:32	-27.5754331	-48.61391827	91
10/04/2021 03:18:32	-27.5754331	-48.61391827	91
10/04/2021 03:28:32	-27.5754331	-48.61391827	91
10/04/2021 04:58:35	-27.5754331	-48.61391827	91
10/04/2021 05:28:35	-27.5747941	-48.61361851	106
10/04/2021 05:38:35	-27.5752035	-48.61474951	90
10/04/2021 05:48:35	-27.5752035	-48.61474951	90
10/04/2021 06:38:31	-27.5747508	-48.6136647	113
10/04/2021 09:38:31	-27.5754331	-48.61391827	91
10/04/2021 11:28:29	-27.5754331	-48.61391827	91
10/04/2021 11:38:29	-27.5747508	-48.61366474	113
10/04/2021 11:48:29	-27.5747508	-48.61366474	113
10/04/2021 11:48:29	-27.5747508	-48.61366474	113
10/04/2021 11:58:29	-27.5754331	-48.61391827	91
10/04/2021 12:08:29	-27.5754331	-48.61391827	91
10/04/2021 12:18:29	-27.5754331	-48.61391827	91
10/04/2021 12:48:29	-27.5747508	-48.61366474	113
10/04/2021 12:58:29	-27.576069	-48.61474422	202
10/04/2021 13:08:35	-27.576069	-48.61474422	202
10/04/2021 13:18:35	-27.576069	-48.61474422	202
10/04/2021 13:28:35	-27.576069	-48.61474422	202
10/04/2021 13:38:35	-27.576069	-48.61474422	202
10/04/2021 13:48:35	-27.576069	-48.61474422	202
10/04/2021 13:58:35	-27.576069	-48.61474422	202

10/04/2021 14:18:35	-27.576069	-48.61474422	202
10/04/2021 14:28:35	-27.576069	-48.61474422	202
10/04/2021 14:38:35	-27.576069	-48.61474422	202
10/04/2021 14:48:35	-27.576069	-48.61474422	202
10/04/2021 14:58:35	-27.576069	-48.61474422	202
10/04/2021 15:08:34	-27.576069	-48.61474422	202
10/04/2021 15:18:34	-27.576069	-48.61474422	202
10/04/2021 15:28:34	-27.576069	-48.61474422	202
10/04/2021 15:38:34	-27.576069	-48.61474422	202
11/04/2021 08:28:31	-23.3483546	-46.86129052	250
11/04/2021 08:38:31	-23.3481898	-46.8609223	250
11/04/2021 08:48:31	-23.3482364	-46.86122262	234
11/04/2021 08:58:31	-23.3486694	-46.86214759	220
11/04/2021 09:08:31	-23.3485556	-46.86197638	185
11/04/2021 09:18:31	-23.3482862	-46.86129016	250
11/04/2021 09:28:31	-23.3482555	-46.86093431	250
11/04/2021 09:38:31	-23.3487058	-46.86195692	190
11/04/2021 09:58:31	-23.3483261	-46.86143186	247
11/04/2021 10:08:31	-23.3482364	-46.86122262	234
11/04/2021 10:28:31	-23.3481898	-46.8609223	250
11/04/2021 10:38:31	-23.3482555	-46.86093431	250
11/04/2021 10:48:31	-23.3482043	-46.86097057	250
11/04/2021 11:08:30	-23.3483261	-46.86143186	247
11/04/2021 11:18:30	-23.3486138	-46.8618225,	184
11/04/2021 12:08:30	-23.3483261	-46.86143186	247
11/04/2021 12:18:30	-23.3483546,	-46.86129052	250
11/04/2021 12:38:30	-23.3483261	-46.86143186	247

11/04/2021 12:48:30	-23.3485789	-46.86187453	167
11/04/2021 12:58:30	-23.3482862	-46.86143403	249
11/04/2021 13:18:30	-23.3483546	-46.86129052	250
11/04/2021 13:28:30	-23.3485495	-46.8617034	185
11/04/2021 13:38:30	-23.348669	-46.86214759	220
11/04/2021 13:58:30	-23.3482862	-46.86143403	249
11/04/2021 14:28:30	-23.3487058	-46.86195692	190
11/04/2021 14:38:30	-23.3485789	-46.86187453	167
11/04/2021 14:48:30	-23.3482862	-46.86143403	249
11/04/2021 14:58:30	-23.3483546	-46.86129052	250
11/04/2021 15:18:30	-23.3482862	-46.86129016	250
11/04/2021 15:28:30	-23.3483546	-46.86129052	250
11/04/2021 15:38:30	-23.3483546	-46.86129052	250
11/04/2021 15:48:30	-23.3483261	-46.86143186	247
11/04/2021 15:58:30	-23.3483261	-46.86143186	247
11/04/2021 16:08:30	-23.3482862	-46.86129016	250
11/04/2021 16:18:30	-23.3483261	-46.86143186	247
11/04/2021 16:28:30	-23.3483694	-46.86130285	250
11/04/2021 16:38:30	-23.3483546	-46.86129052	250
11/04/2021 16:48:30	-23.3482862	-46.86143403	249
11/04/2021 16:58:30	-23.3482862	-46.86143403	249
11/04/2021 17:08:30	-23.3483546	-46.86129052	250
11/04/2021 17:18:30	-23.3483694	-46.86130285	250
11/04/2021 17:28:30	-23.3482862	-46.86129016	250
11/04/2021 17:38:30	-23.3483546	-46.86129052	250
11/04/2021 17:48:30	-23.3482862	-46.86093807	171
12/04/2021 00:28:28	-22.310092	-49.02729257	71

12/04/2021 00:38:28	-22.3100929	-49.02729257	71
12/04/2021 00:48:28	-22.3100929	-49.02729257	71
12/04/2021 00:58:28	-22.3100929	-49.02729257	71
12/04/2021 01:08:28	-22.3100929	-49.02729257	71
12/04/2021 01:18:28	-22.3100929	-49.0272925	71
12/04/2021 01:28:28	-22.3100929	-49.02729257	71
12/04/2021 01:38:28	-22.3100929	-49.02729257	71
12/04/2021 01:48:28	-22.3100929	-49.02729257	71
12/04/2021 01:58:28	-22.3100929	-49.02729257	71
12/04/2021 02:08:27	-22.3100929	-49.02729257	71
12/04/2021 02:18:27	-22.3100929	-49.02729257	71
12/04/2021 02:28:27	-22.3100929	-49.02729257	71
12/04/2021 02:38:27	-22.3100929	-49.02729257	71
12/04/2021 02:48:27	-22.3100929	-49.02729257	71
12/04/2021 02:58:27	-22.3100929	-49.02729257	71
12/04/2021 03:08:27	-22.3100929	-49.02729257	71
12/04/2021 03:18:27	-22.3100929	-49.02729257	71
12/04/2021 03:28:27	-22.3100929	-49.02729257	71
12/04/2021 03:38:27	-22.3100929	-49.02729257	71
12/04/2021 03:48:27	-22.3100929	-49.02729257	71
12/04/2021 03:58:27	-22.3100929	-49.02729257	71
12/04/2021 04:08:27	-22.3100929	-49.02729257	71
12/04/2021 04:18:27	-22.3100929	-49.02729257	71
12/04/2021 04:28:27	-22.3100929	-49.02729257	71
12/04/2021 04:38:27	-22.3100929	-49.02729257	71
12/04/2021 04:48:27	-22.3100929	-49.02729257	71
12/04/2021 04:58:27	-22.3100929	-49.02729257	71

12/04/2021 05:08:27	-22.3100929	-49.02729257	71
12/04/2021 05:18:27	-22.3100929	-49.02729257	71
12/04/2021 05:28:27	-22.3100929	-49.02729257	71
12/04/2021 05:38:27	-22.3100929	-49.02729257	71
12/04/2021 05:48:27	-22.3100929	-49.02729257	71
12/04/2021 05:58:27	-22.3100929	-49.02729257	71
12/04/2021 07:08:27	-22.3262186	-49.09628013	99
12/04/2021 07:38:27	-22.3261729	-49.0962317	97
12/04/2021 07:48:27	-22.3250802	-49.09578775	250
12/04/2021 08:08:27	-22.3250802	-49.09578775	250
12/04/2021 08:18:27	-22.3250802	-49.09578775	250
12/04/2021 08:58:27	-22.3250802	-49.09578775	250
12/04/2021 09:08:27	-22.3262186	-49.09628013	99
12/04/2021 09:28:27	-22.3261729	-49.0962317,	97
12/04/2021 09:38:27	-22.3266237	-49.0967388	71
12/04/2021 09:48:27	-22.3262186	-49.09628013	99

APÊNDICE B - TABELA DE TEMPERATURA DO PRIMEIRO TESTE

Data e Hora	Temperatura (°C)	Data e Hora	Temperatura (°C)
09/04/2021 13:08:29	26.9	10/04/2021 20:48:25	27.5
09/04/2021 13:18:29	26.9	10/04/2021 20:58:25	27.3
09/04/2021 13:28:29	26.9	10/04/2021 21:08:25	27.1
09/04/2021 13:38:29	27	10/04/2021 21:18:25	26.9
09/04/2021 13:48:29	27	10/04/2021 21:28:25	26.7
09/04/2021 13:58:29	27.3	10/04/2021 21:38:25	26.5
09/04/2021 14:08:29	27.4	10/04/2021 21:48:25	26.3
09/04/2021 14:18:29	27.6	10/04/2021 21:58:25	26.1

09/04/2021 14:28:29	28	10/04/2021 22:08:25	26
09/04/2021 14:38:29	28.4	10/04/2021 22:18:25	25.8
09/04/2021 14:48:28	28.3	10/04/2021 22:28:25	25.5
09/04/2021 14:58:28	28.3	10/04/2021 22:38:25	25.3
09/04/2021 15:08:28	28.7	10/04/2021 22:48:25	25.1
09/04/2021 15:18:28	28.5	10/04/2021 22:58:25	24.9
09/04/2021 15:28:28	28.9	10/04/2021 23:08:25	24.8
09/04/2021 15:38:28	30.8	10/04/2021 23:18:25	24.6
09/04/2021 15:48:28	31.5	10/04/2021 23:28:25	24.4
09/04/2021 15:58:28	31	10/04/2021 23:38:25	24.3
09/04/2021 16:08:28	30.8	10/04/2021 23:48:25	24.1
09/04/2021 16:18:28	30.8	10/04/2021 23:58:25	23.9
09/04/2021 16:28:28	30.8	11/04/2021 00:08:35	23.8
09/04/2021 16:38:28	30.6	11/04/2021 00:18:35	23.6
09/04/2021 16:48:28	30.5	11/04/2021 00:28:35	23.5
09/04/2021 16:58:28	30.3	11/04/2021 00:38:35	23.3
09/04/2021 17:08:28	30.1	11/04/2021 00:48:35	23.2
09/04/2021 17:18:28	30.1	11/04/2021 00:58:35	23
09/04/2021 17:28:28	30.1	11/04/2021 01:08:35	22.9
09/04/2021 17:38:28	30.1	11/04/2021 01:18:35	22.8
09/04/2021 17:48:28	30.1	11/04/2021 01:28:35	22.6
09/04/2021 17:58:28	30.1	11/04/2021 01:38:35	22.5
09/04/2021 18:08:27	30.1	11/04/2021 01:48:35	22.3
09/04/2021 18:18:27	30.1	11/04/2021 01:58:35	22.2
09/04/2021 18:38:27	30	11/04/2021 02:08:25	22
09/04/2021 18:48:27	30	11/04/2021 02:18:25	21.9
09/04/2021 18:58:27	29.9	11/04/2021 02:28:25	21.7

09/04/2021 19:08:27	29.9	11/04/2021 02:38:25	21.6
09/04/2021 19:18:27	29.9	11/04/2021 02:48:25	21.5
09/04/2021 19:28:27	29.8	11/04/2021 02:58:25	21.3
09/04/2021 19:38:27	29.8	11/04/2021 03:08:25	21.2
09/04/2021 19:48:27	29.7	11/04/2021 03:18:25	21.1
09/04/2021 19:58:27	29.7	11/04/2021 03:28:25	21
09/04/2021 20:08:27	29.6	11/04/2021 03:38:25	21
09/04/2021 20:18:27	29.5	11/04/2021 03:38:25	20.9
09/04/2021 20:28:27	29.4	11/04/2021 03:58:25	20.7
09/04/2021 20:38:27	29.4	11/04/2021 04:08:25	20.7
09/04/2021 20:48:27	29.3	11/04/2021 04:18:25	20.7
09/04/2021 20:58:27	29.2	11/04/2021 04:28:25	20.6
09/04/2021 22:08:27	28.8	11/04/2021 04:38:25	20.6
09/04/2021 22:18:27	28.7	11/04/2021 04:48:25	20.5
09/04/2021 22:28:27	28.7	11/04/2021 04:58:25	20.5
09/04/2021 22:38:27	28.6	11/04/2021 05:08:25	20.4
09/04/2021 22:48:27	28.4	11/04/2021 05:18:25	20.4
09/04/2021 22:58:27	28.1	11/04/2021 05:28:25	20.5
09/04/2021 23:08:27	27.9	11/04/2021 05:38:25	20.5
09/04/2021 23:18:27	27.7	11/04/2021 05:48:25	20.6
09/04/2021 23:28:27	27.5	11/04/2021 05:58:25	20.7
09/04/2021 23:38:27	27.4	11/04/2021 07:08:25	21.4
09/04/2021 23:48:27	27.3	11/04/2021 07:18:25	21.4
09/04/2021 23:58:27	27.1	11/04/2021 07:28:25	21.3
10/04/2021 00:08:27	27.1	11/04/2021 07:38:25	21.2
10/04/2021 00:18:27	27	11/04/2021 07:48:25	21.1
10/04/2021 00:28:27	27	11/04/2021 07:58:25	21.1

10/04/2021 00:38:27	26.9	11/04/2021 10:08:25	21.5
10/04/2021 00:48:26	26.8	11/04/2021 10:18:25	21.5
10/04/2021 00:58:26	26.8	11/04/2021 10:28:25	21.6
10/04/2021 01:08:26	26.8	11/04/2021 10:38:25	21.6
10/04/2021 01:18:26	26	11/04/2021 10:48:25	21.7
10/04/2021 01:28:26	26	11/04/2021 10:58:25	21.8
10/04/2021 01:38:26	26	11/04/2021 11:08:24	21.9
10/04/2021 01:48:26	26	11/04/2021 11:18:24	22
10/04/2021 01:58:26	26.6	11/04/2021 11:28:24	22
10/04/2021 02:08:26	26.6	11/04/2021 11:38:24	22.1
10/04/2021 02:18:26	26.6	11/04/2021 11:48:24	22.2
10/04/2021 02:28:26	26.6	11/04/2021 11:58:24	22.3
10/04/2021 02:38:26	26.5	11/04/2021 12:08:24	22.4
10/04/2021 02:48:26	26.5	11/04/2021 12:18:24	22.5
10/04/2021 02:58:26	26.5	11/04/2021 12:28:24	22.6
10/04/2021 03:08:26	26.5	11/04/2021 12:38:24	22.7
10/04/2021 03:18:26	26.5	11/04/2021 12:48:24	22.8
10/04/2021 03:28:26	26.4	11/04/2021 12:58:24	22.9
10/04/2021 03:38:26	26.4	11/04/2021 13:08:24	23
10/04/2021 03:48:26	26.4	11/04/2021 13:18:24	23.1
10/04/2021 03:58:26	26.3	11/04/2021 13:28:24	23.2
10/04/2021 04:08:26	26.3	11/04/2021 13:38:24	23.2
10/04/2021 04:18:26	26.2	11/04/2021 13:48:24	23.3
10/04/2021 04:28:26	26.2	11/04/2021 13:58:24	23.4
10/04/2021 04:38:26	26.1	11/04/2021 14:08:24	23.5
10/04/2021 04:48:26	26.1	11/04/2021 14:18:24	23.6
10/04/2021 04:58:26	26	11/04/2021 14:28:24	23.6

10/04/2021 05:08:26	26.1	11/04/2021 14:38:24	23.7
10/04/2021 05:18:26	26	11/04/2021 14:48:24	23.8
10/04/2021 05:28:26	26	11/04/2021 14:58:24	23.9
10/04/2021 05:38:26	26	11/04/2021 15:08:23	23.9
10/04/2021 05:48:26	25.8	11/04/2021 15:18:23	24
10/04/2021 05:58:26	25.6	11/04/2021 15:28:23	24.1
10/04/2021 06:08:25	25.4	11/04/2021 15:38:23	24.1
10/04/2021 06:18:25	25.2	11/04/2021 15:48:23	24.2
10/04/2021 06:28:25	25.1	11/04/2021 15:58:23	24.3
10/04/2021 06:38:25	25.1	11/04/2021 16:08:23	24.4
10/04/2021 06:48:25	25.1	11/04/2021 16:18:23	24.4
10/04/2021 06:58:25	25.1	11/04/2021 16:28:23	24.4
10/04/2021 07:08:24	25.1	11/04/2021 16:38:23	24.5
10/04/2021 07:18:24	25.1	11/04/2021 16:48:23	24.5
10/04/2021 07:28:24	25.1	11/04/2021 16:58:23	24.5
10/04/2021 07:38:24	25.1	11/04/2021 17:08:23	24.6
10/04/2021 07:48:24	25.1	11/04/2021 17:18:25	24.6
10/04/2021 07:58:24	25.1	11/04/2021 17:28:25	24.6
10/04/2021 08:08:24	25	11/04/2021 17:38:25	24.6
10/04/2021 08:18:24	25	11/04/2021 17:48:25	24.6
10/04/2021 08:28:24	24.9	11/04/2021 17:58:25	24.6
10/04/2021 08:38:24	24.9	11/04/2021 22:08:22	24.3
10/04/2021 08:48:24	24.9	11/04/2021 22:18:22	24.2
10/04/2021 08:58:24	24.8	11/04/2021 22:28:22	24.2
10/04/2021 09:08:24	24.8	11/04/2021 22:38:22	24.2
10/04/2021 09:18:24	24.8	11/04/2021 22:48:22	24.1
10/04/2021 09:28:24	24.7	11/04/2021 22:58:22	24.1

10/04/2021 09:38:24	24.7	11/04/2021 23:08:22	24
10/04/2021 09:48:24	24.7	11/04/2021 23:18:22	24
10/04/2021 09:58:24	24.7	11/04/2021 23:28:22	24
10/04/2021 10:08:24	24.7	11/04/2021 23:38:22	23.9
10/04/2021 10:18:24	24.7	11/04/2021 23:48:22	23.9
10/04/2021 10:28:24	24.6	11/04/2021 23:58:22	23.8
10/04/2021 10:38:24	24.6	12/04/2021 00:08:22	23.8
10/04/2021 10:48:24	24.6	12/04/2021 00:18:22	23.8
10/04/2021 10:58:24	24.6	12/04/2021 00:28:22	23.7
10/04/2021 11:08:24	24.6	12/04/2021 00:38:22	23.7
10/04/2021 11:18:24	24.6	12/04/2021 00:48:22	23.6
10/04/2021 11:28:24	24.6	12/04/2021 00:58:22	23.6
10/04/2021 11:38:24	24.6	12/04/2021 01:08:22	23.6
10/04/2021 11:48:24	24.6	12/04/2021 01:18:22	23.7
10/04/2021 11:58:24	24.6	12/04/2021 01:28:22	24
10/04/2021 12:08:24	24.7	12/04/2021 01:38:22	24.2
10/04/2021 12:18:29	24.7	12/04/2021 01:48:22	24.4
10/04/2021 12:28:29	24.7	12/04/2021 01:58:22	24.5
10/04/2021 12:38:29	24.7	12/04/2021 02:08:22	24.6
10/04/2021 12:48:29	24.7	12/04/2021 02:18:21	24.6
10/04/2021 12:58:29	24.8	12/04/2021 02:28:21	24.6
10/04/2021 13:08:29	25.2	12/04/2021 02:38:21	24.6
10/04/2021 13:18:29	25.4	12/04/2021 02:48:21	24.5
10/04/2021 13:28:29	25.5	12/04/2021 02:58:21	24.4
10/04/2021 13:38:29	25.5	12/04/2021 03:08:21	24.4
10/04/2021 13:48:29	25.5	12/04/2021 03:18:21	24.3
10/04/2021 13:58:29	25.5	12/04/2021 03:28:21	24.3

10/04/2021 14:08:29	25.6	12/04/2021 03:38:21	24.2
10/04/2021 14:18:28	25.6	12/04/2021 03:48:21	24.2
10/04/2021 14:28:28	25.6	12/04/2021 03:58:21	24.2
10/04/2021 14:38:28	25.6	12/04/2021 04:08:21	24.2
10/04/2021 14:48:28	25.6	12/04/2021 04:18:21	24.1
10/04/2021 14:58:28	25.6	12/04/2021 04:28:21	24
10/04/2021 15:08:28	25.6	12/04/2021 04:38:21	23.8
10/04/2021 15:18:28	25.6	12/04/2021 04:48:21	23.7
10/04/2021 15:28:28	25.6	12/04/2021 04:58:21	23.5
10/04/2021 15:38:28	25.6	12/04/2021 05:08:21	23.4
10/04/2021 15:48:28	25.8	12/04/2021 05:18:21	23.3
10/04/2021 15:58:28	26	12/04/2021 05:28:21	23.2
10/04/2021 16:08:25	26.4	12/04/2021 05:38:21	23.1
10/04/2021 16:18:25	26.6	12/04/2021 05:48:21	23.1
10/04/2021 16:28:25	26.8	12/04/2021 05:58:21	23
10/04/2021 16:38:25	27.1	12/04/2021 06:08:21	22.9
10/04/2021 16:48:25	27.4	12/04/2021 06:18:21	22.8
10/04/2021 16:58:25	27.7	12/04/2021 06:28:21	22.6
10/04/2021 17:08:25	27.9	12/04/2021 06:38:20	22.5
10/04/2021 17:18:25	28.2	12/04/2021 06:48:20	22.4
10/04/2021 17:28:25	28.4	12/04/2021 06:58:20	22.4
10/04/2021 17:38:25	28.6	12/04/2021 07:08:20	22.8
10/04/2021 17:48:25	28.7	12/04/2021 07:18:20	23.1
10/04/2021 17:58:25	28.8	12/04/2021 07:28:20	23.3
10/04/2021 18:08:25	28.9	12/04/2021 07:38:20	23.5
10/04/2021 18:18:25	28.9	12/04/2021 07:48:20	23.7
10/04/2021 18:28:25	29	12/04/2021 07:58:20	23.8

10/04/2021 18:38:25	29	12/04/2021 08:08:20	23.8
10/04/2021 18:48:25	29	12/04/2021 08:18:20	23.8
10/04/2021 18:58:25	29.1	12/04/2021 08:28:20	24.1
10/04/2021 19:08:25	29.1	12/04/2021 08:38:20	24.3
10/04/2021 19:18:25	29.1	12/04/2021 08:48:20	24.4
10/04/2021 19:28:25	29	12/04/2021 08:58:20	24.5
10/04/2021 19:38:25	28.9	12/04/2021 09:08:20	24.5
10/04/2021 19:48:25	28.7	12/04/2021 09:18:20	24.6
10/04/2021 19:58:25	28.5	12/04/2021 09:28:20	24.6
10/04/2021 20:08:25	28.3	12/04/2021 09:38:20	24.8
10/04/2021 20:18:25	28.1	12/04/2021 09:48:20	24.8
10/04/2021 20:28:25	27.9	12/04/2021 09:58:20	25
10/04/2021 20:38:25	27.7	--	--

APÊNDICE C - TABELA DE LOCALIZAÇÃO DO SEGUNDO TESTE

Data e Hora	Latitude	Longitude	Raio (m)
13/04/2021 01:58:12	-22.3426171	-49.08840661	108
13/04/2021 02:28:12	-22.3186114	-49.04978312	93
13/04/2021 04:28:12	-21.3675056	-50.28371356	65
13/04/2021 04:58:12	21.2127173	-50.42791367	101
13/04/2021 05:18:12	-21.2045345	-50.51406651	114
13/04/2021 05:28:12	-21.2045345	-50.51406651	114
13/04/2021 05:38:12	-21.2045345	-50.51406651	114
13/04/2021 05:48:12	-21.2045345	-50.51406651	114
13/04/2021 05:58:12	-21.2045345	-50.51406651	114
13/04/2021 06:08:12	-21.2045345	-50.51406651	114
13/04/2021 06:18:12	-21.2045345	-50.51406651	114

13/04/2021 06:28:12	-21.2045345	-50.51406651	114
13/04/2021 06:38:12	-21.2242434	-50.45448451	56
13/04/2021 06:48:12	-21.0480452	-49.37897668	90
13/04/2021 09:38:12	-20.7976711	-49.36772442	49
13/04/2021 09:48:12	-20.8176549	-49.38175672	61
13/04/2021 10:08:12	-20.8220284	-49.38424157	75
13/04/2021 10:18:12	-20.8066587	-49.37681191	124
13/04/2021 10:38:12	-20.8219274	-49.37781725	89
13/04/2021 10:58:12	-20.8239619	-49.40149454	78

APÊNDICE B - TABELA DE TEMPERATURA DO SEGUNDO TESTE

Data e Hora	Temperatura (°C)	Data e Hora	Temperatura (°C)
13/04/2021 00:08:16	27.5	13/04/2021 05:18:15	26.8
13/04/2021 00:18:16	27.5	13/04/2021 05:28:15	26.6
13/04/2021 00:28:16	27.5	13/04/2021 05:38:15	26.5
13/04/2021 00:38:16	27.5	13/04/2021 05:48:15	26.5
13/04/2021 00:48:16	27.5	13/04/2021 05:58:15	26.6
13/04/2021 00:58:16	27.5	13/04/2021 06:08:14	26.7
13/04/2021 01:08:16	27.5	13/04/2021 06:18:14	26.8
13/04/2021 01:18:16	27.5	13/04/2021 06:28:14	26.8
13/04/2021 01:28:16	27.5	13/04/2021 06:38:14	26.8
13/04/2021 01:38:16	27.5	13/04/2021 06:48:14	26.9
13/04/2021 01:48:16	27.5	13/04/2021 06:58:14	26.9
13/04/2021 01:58:16	27.5	13/04/2021 07:08:14	27
13/04/2021 02:08:14	27.7	13/04/2021 07:18:14	27.1
13/04/2021 02:18:14	27.6	13/04/2021 07:28:14	27.1
13/04/2021 02:28:14	27.3	13/04/2021 07:38:14	27.2

13/04/2021 02:38:14	27.1	13/04/2021 07:48:14	27.3
13/04/2021 02:48:14	27	13/04/2021 07:58:14	27.4
13/04/2021 02:58:14	27	13/04/2021 08:08:14	27.6
13/04/2021 03:08:14	27	13/04/2021 08:18:14	27.8
13/04/2021 03:18:14	27	13/04/2021 08:28:14	28
13/04/2021 03:28:14	27.1	13/04/2021 08:38:14	28.1
13/04/2021 03:48:14	27.2	13/04/2021 08:48:14	28.3
13/04/2021 03:58:14	27.3	13/04/2021 08:58:14	28.5
13/04/2021 04:08:15	27.3	13/04/2021 09:08:14	28.7
13/04/2021 04:18:15	27.3	13/04/2021 09:18:14	28.9
13/04/2021 04:28:15	27.3	13/04/2021 09:28:14	29
13/04/2021 04:38:15	27.3	13/04/2021 09:38:14	29.2
13/04/2021 04:48:15	27.3	13/04/2021 09:48:14	29.4
13/04/2021 04:58:15	27.3	13/04/2021 09:58:14	29.7
13/04/2021 05:08:15	27.1	--	--