

**UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS**

Lucas Sossai

**Sistema de tracking de animais utilizando tecnologia
iBeacon**

**São Carlos
2019**

Lucas Sossai

**Sistema de tracking de animais utilizando tecnologia
iBeacon**

Monografia apresentada ao Curso de Engenharia Elétrica com Ênfase em Eletrônica, da Escola de Engenharia de São Carlos da Universidade de São Paulo, como parte dos requisitos para obtenção do título de Engenheiro Eletricista.

Orientador: Prof. Dr. Eduardo do Valle Simões

**São Carlos
2019**

AUTORIZO A REPRODUÇÃO TOTAL OU PARCIAL DESTE TRABALHO,
POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO, PARA FINS
DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Ficha catalográfica elaborada pela Biblioteca Prof. Dr. Sérgio Rodrigues Fontes da
EESC/USP com os dados inseridos pelo(a) autor(a).

S933s Sossai, Lucas
 Sistema de tracking de animais utilizando a
 tecnologia iBeacon / Lucas Sossai; orientador Eduardo
 do Valle Simões. São Carlos, 2019.

Monografia (Graduação em Engenharia Elétrica com
ênfase em Eletrônica) -- Escola de Engenharia de São
Carlos da Universidade de São Paulo, 2019.

1. Internet das Coisas. 2. LoRa. 3. Bluetooth Low
Energy. 4. Monitoramento de Animais. I. Título.

FOLHA DE APROVAÇÃO

Nome: Lucas Sossai

Título: "Sistema de tracking de animais utilizando tecnologia iBeacon"

Trabalho de Conclusão de Curso defendido e aprovado

em 6/6/2019,

com NOTA 8,0 (oito, zero), pela Comissão Julgadora:

Prof. Dr. Eduardo do Valle Simões - Orientador - SSC/ICMC/USP

Prof. Dr. Fernando Santos Osório - SSC/ICMC/USP

Prof. Dr. Maximiliam Luppe - SEL/EESC/USP

Coordenador da CoC-Engenharia Elétrica - EESC/USP:
Prof. Associado Rogério Andrade Flauzino

Este trabalho é dedicado à minha família.

AGRADECIMENTOS

Em primeiro lugar, gostaria de agradecer à minha família por todo incentivo e apoio nos momentos críticos.

A minha namorada Milena pela compreensão e apoio durante todo o período dedicado aos estudos.

Também gostaria de agradecer meus grandes irmãos da República Poltergeist, por me acolherem durante boa parte da graduação e por tudo que aprendemos juntos durante esses anos.

Agradeço também o João Fernando Calcagno Camargo, por ter desenvolvido e providenciado o *beacon* utilizado no projeto.

Por fim, agradeço ao Professor Eduardo do Valle Simões por me orientar nesse projeto.

RESUMO

Um problema atualmente encontrado em fazendas é a falta de algumas informações sobre seus animais. O objetivo desse trabalho consiste em criar um sistema de comunicação entre sistemas embarcados para localização de animais e monitoramento de seus dados visando um controle melhor do seu rebanho e análise de dados que podem ajudar a decidir melhores soluções para o produtor. O sistema desenvolvido foi feito a partir de *beacons*, pequenos dispositivos de baixo consumo de energia que enviam sinal via *Bluetooth*. Com eles acoplados aos animais, é possível localizá-los com microcontroladores de baixo consumo de energia que são responsáveis pela leitura dos sinais dos *beacons* e enviar eles via *LoRa* para uma central. A função desta é de processar os dados e enviá-los para um banco de dados.

Palavras-chave: Internet das Coisas, LoRa, Bluetooth Low Energy, Monitoramento de Animais.

LISTA DE FIGURAS

Figura 1 – Estrutura de uma mensagem BLE.	14
Figura 2 – Protocolo iBeacon dentro da estrutura da mensagem BLE.	15
Figura 3 – Diagrama comparativo dos principais métodos de comunicação.	17
Figura 4 – Arquitetura <i>publish/subscribe</i> no protocolo MQTT	18
Figura 5 – (Da esquerda pra direita) Lilypad, Sparkfun Pro Micro, Arduino Mega	18
Figura 6 – Página principal da ferramenta Node-RED	19
Figura 7 – Tag beacon Animalltag com sua proteção plástica	20
Figura 8 – Módulo <i>Heltec WiFi LoRa 32</i>	21
Figura 9 – <i>Raspberry Pi 3 Model B</i>	22
Figura 10 – Tela inicial do phpMyAdmin	24
Figura 11 – Diagrama do sistema	25
Figura 12 – Fluxograma do software da estação coletora.	26
Figura 13 – Exemplo de pacote de dados enviado via <i>LoRa</i>	27
Figura 14 – Fluxograma do software da estação central.	27
Figura 15 – Flow principal do Node-RED.	28
Figura 16 – Estrutura da tabela blescan no servidor Apache.	29
Figura 17 – Estação coletora utilizada nos testes.	30
Figura 18 – Mapa elaborado no Google Earth com os pontos onde foram realizados os testes.	31
Figura 19 – Perfil de elevação do ambiente de testes.	32
Figura 20 – Estação coletora enviando dados no Ponto D.	33

LISTA DE TABELAS

Tabela 1 – Característica do do microcontrolador EM6819	20
Tabela 2 – Característica do chip <i>bluetooth</i> EM9301	21
Tabela 3 – Especificações técnicas do módulo <i>Heltec WiFi LoRa 32</i>	22
Tabela 4 – Esepcificações técnicas da <i>Raspberry Pi 3 Model B</i>	23
Tabela 5 – Tabela de resultados da taxa de perda de pacotes.	33
Tabela 6 – Características e resultados dos pontos de teste	34

LISTA DE ABREVIATURAS E SIGLAS

IoT	Internet of Things
BLE	Bluetooth Low Energy
RFID	Radio-Frequency IDentification
PIB	Produto Interno Bruto
PoE	Power-over-Ethernet
UUID	Universally Unique Identifier
RF	Rádio Frequênciа
CRC	Cyclic Redundancy Check
SSID	Service Set IDentifier
MAC	Media Access Control
MQTT	Message Queue Telemetry Transport
M2M	Machine-To-Machine
IDE	Integrated Development Environment

SUMÁRIO

1	INTRODUÇÃO	12
1.1	Contextualização	12
1.2	Objetivos	12
1.3	Organização do trabalho	13
2	REVISÃO BIBLIOGRÁFICA E METODOLOGIA	14
2.1	Bluetooth Low Energy	14
2.2	<i>Beacons</i>	14
2.2.1	iBeacon	14
2.2.2	Eddystone	15
2.3	LoRa	16
2.4	<i>Message Queue Telemetry Transport(MQTT)</i>	17
2.5	Arduino	18
2.6	Node-RED	19
2.7	Beacon AnimalITag	20
2.8	Módulo <i>Heltec WiFi LoRa 32</i>	21
2.9	Raspberry Pi	22
2.10	Sistema de gerenciamento dos dados	23
2.10.1	Servidor Apache	23
2.10.2	MySQL	23
2.10.3	phpMyAdmin	23
3	DESENVOLVIMENTO	25
3.1	Projeto	25
3.2	Desenvolvimento dos módulos <i>Heltec WiFi LoRa 32</i>	26
3.2.1	Estação coletora	26
3.2.2	Comunicação entre os módulos	26
3.2.3	Estação central	27
3.3	Desenvolvimento da <i>Raspberry Pi</i>	28
3.3.1	Mosquitto	28
3.3.2	Node-RED	28
3.3.3	Banco de Dados no servidor Apache	29
4	EXPERIMENTOS E RESULTADOS	30
4.1	Estação coletora	30
4.2	Ambiente de testes	31

4.3	Experimento 1: Avaliação da taxa de perda de pacotes	32
4.3.1	Resultados	32
4.4	Experimento 2: Avaliação da velocidade de transmissão de pacotes	33
4.4.1	Resultado	33
5	CONCLUSÃO	35
5.1	Resumo	35
5.2	Vantagens e Desvantagens	35
5.3	Relacionamento entre o Curso e o Projeto	35
5.4	Trabalhos Futuros	35
	REFERÊNCIAS	37

1 INTRODUÇÃO

1.1 Contextualização

A Internet das Coisas (IoT) é um novo paradigma que vem rapidamente ganhando terreno no cenário das telecomunicações modernas sem fio, a ideia básica deste conceito é a presença pervasiva em torno de nós de uma variedade de coisas ou objetos - como *beacons*, identificação por radiofrequência (RFID), sensores, atuadores, telefones celulares, etc ([ATZORI, 2010](#)). Com o surgimento de placas de desenvolvimento nos últimos anos como a Raspberry Pi, que possui um microprocessador como componente principal, e a plataforma Arduino, que possui microcontroladores como base, se tornou mais acessível prototipar projetos de diversas naturezas, como automação residencial, robótica e projetos em IoT.

Em 2018, a pecuária no Brasil foi responsável de movimentar 597,22 bilhões de reais, representando 7,7% do PIB Brasileiro ([ABIEC, 2018](#)), em razão disso, busca-se cada vez mais alternativas que visam melhorar a produtividade e qualidade desse setor. Diversos problemas ligados à falta de localização e contagem dos animais podem ocorrer, como o risco de o animal ir para um lugar onde não deveria ou de não saber o número correto de quantos animais possui em cada área de forma imediata. A análise de variáveis em campo é feita esporadicamente e com intervalos longos, pois no campo a variação espacial e temporal é lenta e possui uma gama muito baixa de variáveis se comparada à indústria, trazendo resultados inexatos em razão do processo ser suscetível ao erro humano e limitado em relação à quantidade e periodicidade dos dados coletados ([ALBUQUERQUE, 2009](#)). O monitoramento constante permite analisar dados sobre o comportamento dos animais, como em qual horário eles se alimentam e quais regiões eles frequentam durante o dia. Diante desse contexto, a motivação principal por trás desse trabalho é ajudar os profissionais que trabalham com animais em suas fazendas a conseguirem ter um melhor controle sobre seu rebanho de forma automatizada e com a possibilidade de colher esses dados distantes da sede da fazenda tendo a tecnologia dos *beacons* para rastreamento dos animais e o protocolo *LoRa* para transmissão de dados em longas distâncias.

1.2 Objetivos

O projeto tem como função desenvolver, implementar e testar um sistema de localização de animais em fazendas baseado em módulos *Heltec WiFi LoRa 32*. Os principais objetivos são configurar os módulos *Heltec WiFi LoRa 32* para identificar os sinais Bluetooth vindos dos beacons, transmitir esses dados via LoRa para um receptor, e à partir dele enviar para um servidor baseado em Raspberry Pi, responsável por ser

a central de informações, salvando os resultados coletados em um banco de dados local para análise. Para validação do sistema os testes apresentam os resultados de perda de velocidade de transmissão para distância máxima de 1 Km de comunicação via *LoRa* esperada entre os dispositivos.

1.3 Organização do trabalho

No Capítulo 2 é apresentada a revisão bibliográfica dos principais conceitos utilizados durante o desenvolvimento do trabalho e quais foram as ferramentas, utilitários e dispositivos de hardware usados no projeto. O Capítulo 3 descreve em detalhes como o projeto foi estruturado, desenvolvido e os testes feitos. No capítulo 4 são apresentados os resultados obtidos e as dificuldades e limitações do projeto. Finalmente no Capítulo 5 é apresentado a conclusão além de ideias para trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA E METODOLOGIA

2.1 Bluetooth Low Energy

O BLE é uma tecnologia de rede sem fio concebida e comercializada pela *Bluetooth Special Interest Group*. Em comparação com o Bluetooth clássico, o BLE foi melhorado para diminuir custo e consumo energético além de contar com um novo protocolo de criptografia de 128 bits. Por sua característica principal ser a economia de energia, um dispositivo BLE permanece em modo *sleep* durante maior parte do tempo. Sendo assim, essa tecnologia foi desenvolvida para aplicações que precisam enviar poucas informações, com velocidade máxima de transmissão de 1 Mbps, menor do que o Bluetooth Clássico de 3 Mbps. O baixo consumo energético, com picos de 6 mA e média de 1 μ A, permite que dispositivos com essa tecnologia possam ser alimentados por baterias comuns como pilhas convencionais com autonomia de semanas até meses. Para o futuro, é previsto que o BLE esteja presente em mais de 1 bilhão de dispositivos devido ao crescente uso da tecnologia do Bluetooth Clássico, utilizada em celulares, carros, notebooks, e diversos outros dispositivos. Esta popularização do Bluetooth Clássico serve como uma espécie de alavanca para impulsionar o crescimento da tecnologia do BLE ([GOMEZ; BOSCH; PARADELLS, 2012](#)).

A figura 1 ilustra a estrutura de um pacote de mensagem enviado por um dispositivo BLE, com seu tamanho podendo variar de 8 à 47 bytes :

Figura 1: Estrutura de uma mensagem BLE.

Preambulo (1 byte)	Endereço de acesso (4 bytes)	PDU Cabeçalho (2 bytes)	PDU carga útil (37 bytes)	CRC (3 bytes)
-----------------------	---------------------------------	----------------------------	------------------------------	------------------

Fonte: Elaborado pelo autor

2.2 Beacons

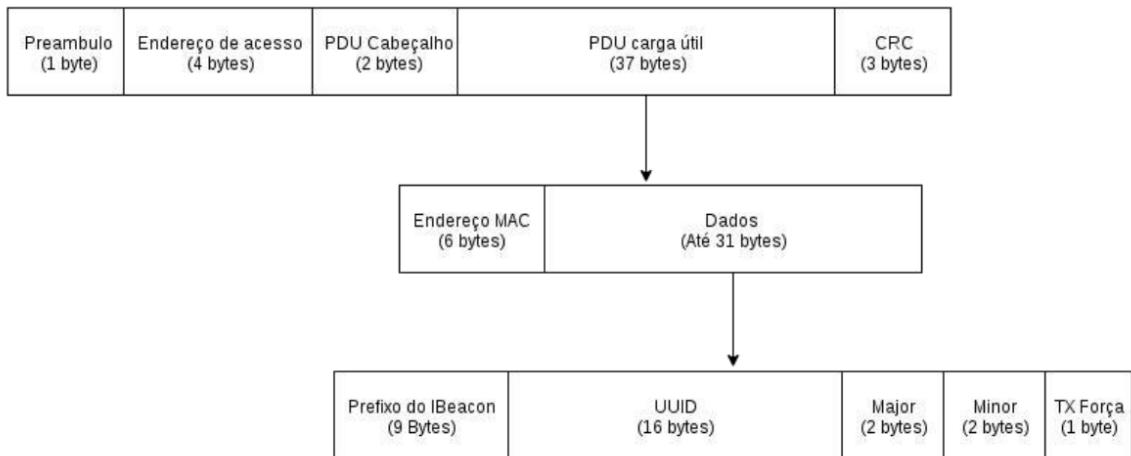
Um *beacon* é um dispositivo que funciona como *tag* por meio da tecnologia BLE, transmitindo mensagem em *broadcast* para todos os dispositivos que estejam em um raio de cerca de 100 metros e possuam a função BLE ativada ([ZAFARI; PAPAPANAGIOTOU, 2015](#)). Abaixo são listados dois dos principais protocolos:

2.2.1 iBeacon

O conceito de *beacon* ficou mais popular após o aparecimento do iBeacon, criado em 2013 pela Apple. Trata-se de um protocolo específico baseado na tecnologia BLE e se

destina a ser usado como um produto voltado para os dispositivos da empresa ([ZAFARI; PAPAPANAGIOTOU, 2015](#)). A figura 2 ilustra como o protocolo desenvolvido pela Apple se relaciona com a estrutura de uma mensagem BLE. Dentro da carga útil, o MAC contém a identificação única do transmissor e os 31 bytes restantes são utilizados para transmitir os dados. A estrutura do pacote iBeacon representada por último ocupa 30 desses 31 bytes disponíveis.

Figura 2: Protocolo iBeacon dentro da estrutura da mensagem BLE.



Fonte: Elaborado pelo autor.

Os 4 principais componentes da estrutura ibeacon são:

- **UUID:** O Identificador Universal Exclusivo é um identificador único, específico a uma empresa, um aplicativo ou alguém que possui um ou mais *beacons* e devem estar no seguinte padrão: 0A22AE44-6B9D-4C76-8884-73F311FB3F57
- **Major:** É utilizado para criar grupos de beacons sob o mesmo UUID. Deve estar definido entre 1 e 65535.
- **Minor:** Combinado com o major, permite refinar a consolidação do beacon. Também deve estar definido entre 1 e 65535.
- **Tx Força:** Valor identificando a potência de transmissão, normalmente é utilizado para estimar a distância do *tag* ao receptor BLE.

2.2.2 Eddystone

Trata-se de um projeto de código aberto desenvolvido pela Google que funciona para Android e iOS. O protocolo Eddystone inclui diferentes tipos padrões, adequados para diversos tipos de implementações ([DEUGO, 2016](#)). Os principais tipos de protocolos são:

- **Eddystone-UID:** equivalente ao pacote iBeacon, com um ID específico para identificar o dispositivo e acionar ações em seu aplicativo.
- **Eddystone-URL:** este pacote de informações envia uma URL diretamente para o celular. Isso permite que um usuário que possua algum aplicativo capaz de interpretar essa mensagem, como o Google Chrome ou o Samsung Internet Browser, receba uma notificação, visualize informações e interaja com aplicações, sem a necessidade de instalar aplicativos específicos.
- **Eddystone-TLM:** permite que informações de telemetria com o nível de bateria ou dados de sensores sejam enviadas.

2.3 LoRa

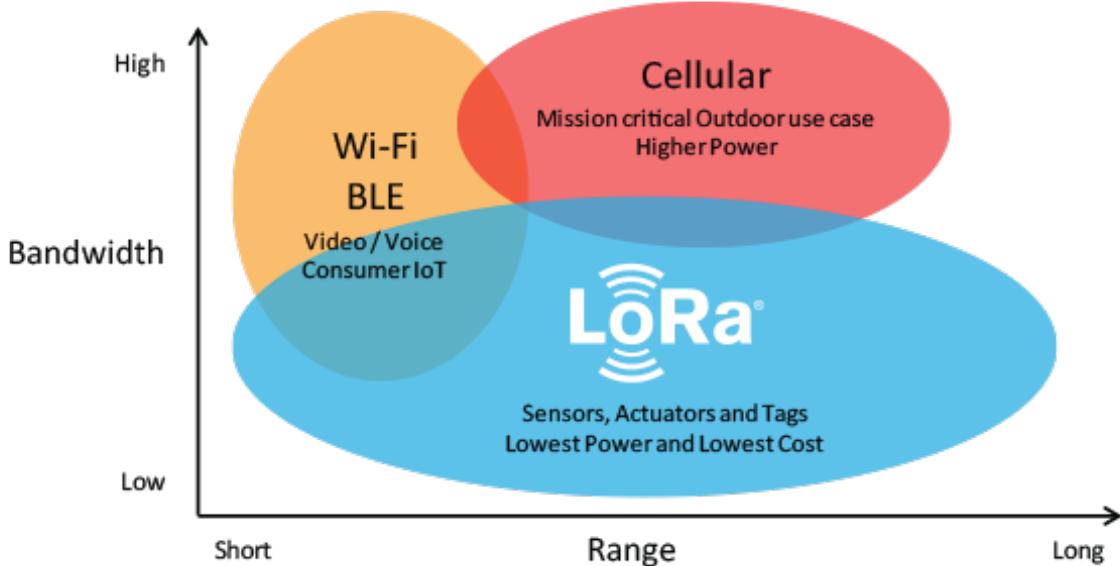
LoRa é um padrão físico de modulação de sinal desenvolvido para comunicações de longo alcance e baixo consumo de energia criada pela empresa Semtech. Devido à seu baixo consumo de energia, esta tecnologia vem sendo integrada em dispositivos IoT para aplicações de iluminação pública, Agricultura de Precisão, dispositivos vestíveis, entre outros (GOULART, 2018).

No Brasil, é possível operar o protocolo LoRa na frequência de 915 MHz, dentro das faixas de frequências não licenciadas de 902 à 907.5 MHz e 915 à 928 MHz. O uso da faixa de frequência não licenciada traz o benefício de reduzir os custos de implantação, permitindo que qualquer pessoa possa utilizar, que também causa a principal desvantagem, que é a interferência de outros sinais (REYNDERS; MEERT; POLLIN, 2016).

O protocolo LoRa adota dois mecanismos para melhorar a qualidade de transmissão e reduzir a taxa de pacotes perdidos. O primeiro mecanismo é a técnica de correção de erros conhecida como *Cyclic Redundancy Check* (CRC), que é aplicado em cada pacote de mensagem enviado. É possível configurar o CRC no LoRa por meio do parâmetro *code rate* (CR), que diminui a taxa de pacotes perdidos quanto maior for o CR. É importante ressaltar que esses parâmetros também impactam o tempo total de transmissão, de forma que para se obter uma transmissão mais robusta é necessário um maior tempo de transmissão e consequentemente menor a velocidade de dados enviados (AYELE, 2017).

O segundo mecanismo utilizado é a técnica de modulação conhecida como *Chirp Spread Spectrum* (CSS), que tem a função de codificar o pacote de dados a ser enviado e o dividir em vários pedaços que vão ser transmitidos separadamente em todo espectro de frequência da largura de banda disponível (REYNDERS; POLLIN, 2016). A figura 3 ilustra como o protocolo de comunicação LoRa se compara em relação aos outros protocolos.

Figura 3: Diagrama comparativo dos principais métodos de comunicação.



Fonte: ([SEMTECH, 2019](#))

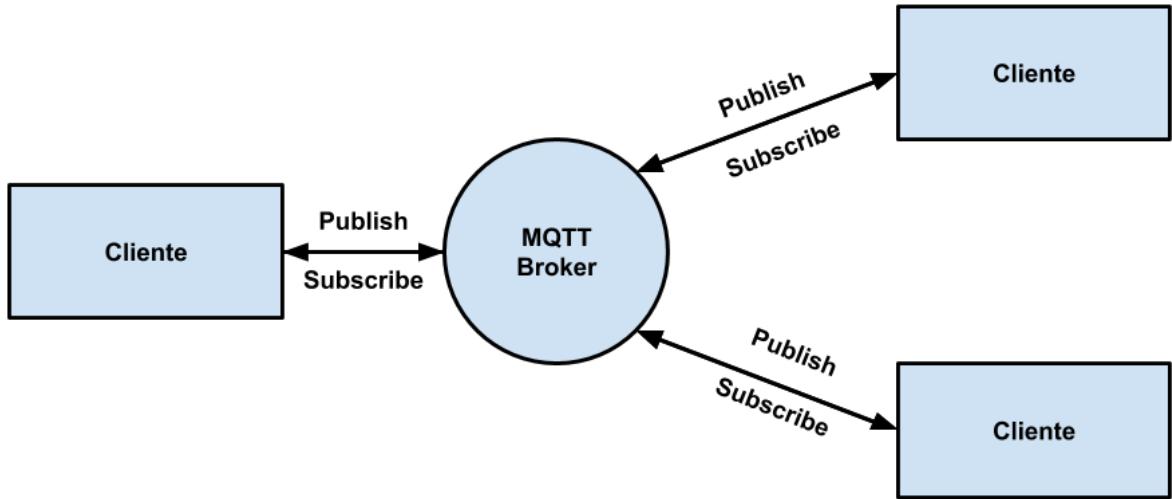
2.4 ***Message Queue Telemetry Transport(MQTT)***

O protocolo de comunicação MQTT foi desenvolvido originalmente em 1999 pela empresa IBM com o propósito de ser extremamente leve e rápido e é utilizado principalmente na comunicação *machine-to-machine* (M2M)([LAMPKIN et al., 2012](#)). Os principais benefícios do protocolo MQTT para sistemas embarcados são ([LOCKE, 2010](#)):

- Baixo custo computacional
- Cabeçalho de pacote reduzido, reduzindo o tamanho das mensagens a serem trocadas
- Sistema de troca de mensagens assíncrono
- Escalabilidade, as mensagens são publicadas em tópicos e os clientes utilizam do padrão *publish/subscribe* na comunicação

A figura 4 ilustra a arquitetura *publish/subscribe*, com um servidor central nomeado *broker*. A comunicação se dá em volta dos tópicos, onde os clientes conectados podem tanto publicar informações nos tópicos desejados quanto se subscrever e esperar chegar dados de outros clientes nos tópicos subscritos. Tanto *publishers* como *subscribers* são clientes no protocolo MQTT e conectam-se a apenas um broker. No entanto, um mesmo cliente MQTT pode ser *publisher* e *subscriber* em diversos tópicos.([MELO et al., 2017](#))

Figura 4: Arquitetura *publish/subscribe* no protocolo MQTT



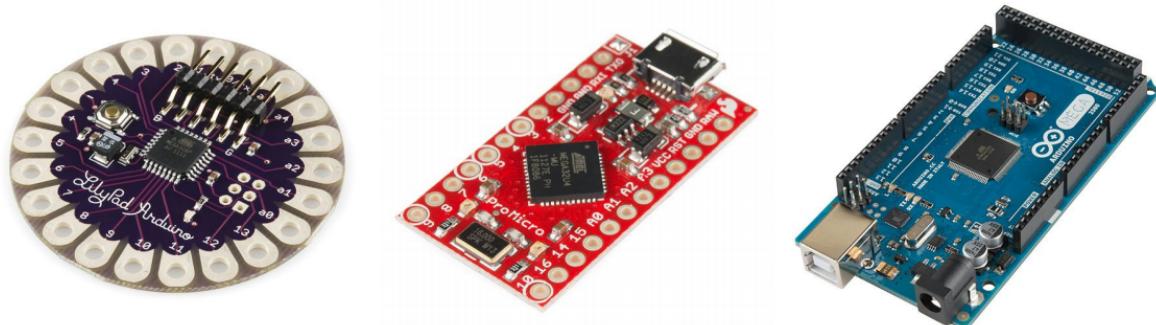
Fonte: Elaborado pelo autor.

2.5 Arduino

Um dos grandes problemas encontrados quando vamos realizar um experimento prático em um projeto de alta complexidade é o tempo e conhecimento necessário para realizá-lo. Uma das formas de contornar isso é utilizando plataformas de desenvolvimento que, ao permitirem uma rápida prototipagem, estas plataformas podem ser pensadas como ferramenta de projeto e de aprendizado.(FONSECA; VEGA, 2011)

Arduino é uma empresa *open-source* de hardware e software para prototipagem eletrônica. O projeto *Arduino Community* se refere à comunidade que desenvolve e utiliza plataformas de desenvolvimento baseadas em microcontroladores (ARDUINO, 2015). Essas plataformas de desenvolvimento são conhecidas como módulos *Arduino* e alguns exemplos estão ilustrados na figura 5.

Figura 5: (Da esquerda pra direita) Lilypad, Sparkfun Pro Micro, Arduino Mega



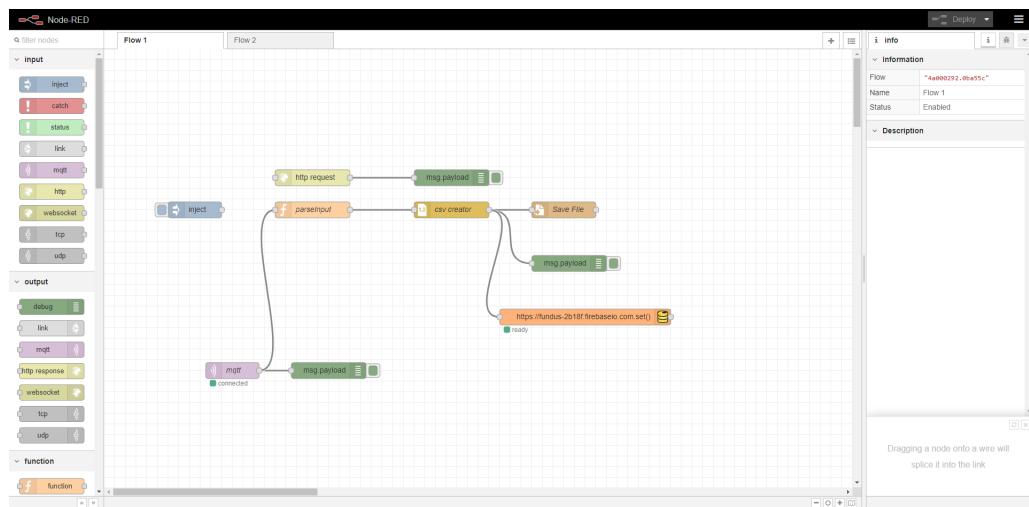
Fonte: (ARDUINO, 2015)

Para o desenvolvimento dos softwares, é disponibilizado uma *Integrated Development Environment* (IDE) própria, nomeada *Arduino Software* e que utiliza da linguagem de programação C. Muitas bibliotecas são disponibilizadas gratuitamente graças à comunidade *open-source*, agilizando o desenvolvimento dos projetos.

2.6 Node-RED

O Node-RED é uma ferramenta de programação de software voltada para sistemas embarcados e IoT construída em Node.js ([SISINNI et al., 2018](#)). A ferramenta fornece um editor gráfico a fim de conectar hardware, *Application Programming Interfaces* (APIs) e serviços *online* como parte da Internet das Coisas ([BLACKSTOCK; LEA, 2014](#)). A figura 6 mostra a interface principal da ferramenta, com o diretório de nós a esquerda, o editor de *flows* no centro e a aba de informação sobre os nós à direita.

Figura 6: Página principal da ferramenta Node-RED



Fonte: Elaborado pelo autor

Os principais nós utilizados nos projetos são:

- **Inject:** Utilizado para injetar alguma informação no sistema quando pressionada ou pode funcionar em modo *loop*. Normalmente é utilizada para fazer testes e checar se o sistema está se comportando como o esperado.
- **Mqtt:** Utilizado para configurar o cliente Mqtt à conectar no *broker* e selecionar quais tópicos quer subscrever ou publicar informações.
- **Debug:** Utilizado para verificar os dados que estão fluindo e conferir o funcionamento do software.
- **Function:** Esse nó é utilizado quando precisamos escrever uma função específica em JavaScript.

2.7 Beacon AnimalITag

O *beacon* utilizado no projeto foi criado pela AnimalITag e é baseado no chip EM9301, da fabricante suíça EM Electronics. O chip escolhido cumpre as normas *Bluetooth* 4.0 e é negociado a um bom preço, deixando o custo do produto final adequado aos objetivos comerciais da empresa. Existem duas versões do tag, uma com acelerômetro que transmite a quantidade de movimento detectado no último minuto mas que não foi utilizada no projeto, e uma simples onde só é enviado a identificação utilizando o protocolo iBeacon. Na figura 7 é possível observar o *beacon*.

Figura 7: Tag beacon AnimalITag com sua proteção plástica



Fonte: ([HELTEC, 2018](#)).

O circuito do *beacon* utilizado é composto por um microcontrolador EM6819, um chip *bluetooth* EM9301, suporte para bateria CR2032, botão liga/desliga, uma antena desenhada como trilha na própria PCB fora os componentes necessários para o funcionamento dos elementos de processamento e transmissão ([CAMARGO, 2018](#)). O software embarcado no *beacon* está configurado para enviar os pacotes iBeacon a cada um segundo e nesse modo de funcionamento a autonomia da bateria é de 15 dias. As especificações técnicas do microcontrolador e chip *bluetooth* utilizados estão nas tabelas 1 e 2

Tabela 1: Característica do do microcontrolador EM6819

Arquitetura	8-bit RISC
Comunicação	SPI e UART
Memória Flash	18 Kbytes
Memória RAM	512 Kbytes
Corrente	140 µA
Tensão de Alimentação	0.9 - 3.6 V DC

Fonte: ([CAMARGO, 2018](#)).

Tabela 2: Característica do chip *bluetooth* EM9301

Tecnologia	Bluetooth 4.0
Frequência RF	2.4 GHz
Potência Máxima	+4dBm
Corrente	12mA (transmitindo em 0dBm)
Tensão de alimentação	3.3 V DC
Distância	40 metros (visada direta)

Fonte: ([CAMARGO, 2018](#)).

2.8 Módulo *Heltec WiFi LoRa 32*

O módulo *Heltec WiFi LoRa 32* ilustrado na figura 8 desenvolvido pela Heltec Automation é uma plataforma de desenvolvimento baseada no microcontrolador ESP32 e que já possui integrado o chip SX1276 para comunicação LoRa.

Figura 8: Módulo *Heltec WiFi LoRa 32*



Fonte: ([HELTEC, 2018](#)).

Os principais motivos que levaram ao uso do módulo no projeto foi a facilidade de implementação pois é possível desenvolver na plataforma Arduino, com acesso à bibliotecas que aceleram a prototipação. Além disso, o módulo tem a capacidade de utilizar 3 tipos de comunicação sem fio: WiFi, Bluetooth e LoRa, tornando-se uma ferramenta muito útil no desenvolvimento de projetos IoT e atendendo aos requisitos técnicos do projeto sem a necessidade de componentes externos. As especificações técnicas do módulo estão na tabela 3.

Tabela 3: Especificações técnicas do módulo *Heltec WiFi LoRa 32*

Microcontrolador	ESP32 (240 MHz, Wi-Fi, Bluetooth 4.2 e BLE)
Chip LoRa	SX1276
Potência máxima LoRa	$18dB \pm 2dB$
Distância máxima LoRa	3,5 Km
Recursos do Hardware	UART x 3; SPI x 2; I2C x 2; I2S x 1
	ADC 12 bits; DAC 8 bits
	29 GPIO's
Display	OLED 128*64 de 0.96 polegadas
FLASH	4MB SPI
Interface	Micro USB x 1; Conector IPEX para antena LoRa x 1
Tamanho	50.2 x 25.5 x 9.74 mm

Fonte: ([HELTEC, 2018](#)).

2.9 Raspberry Pi

A Raspberry Pi é um computador de baixo custo com o tamanho de um cartão de crédito e foi desenvolvido no Reino Unido pela Fundação Raspberry Pi ([RASPBERRYPI, 2019](#)). O propósito dessa ferramente é oferecer uma alternativa barata, prática e acessível para que pessoas possam explorar melhor a computação. A figura 9 mostra o módulo Raspberry Pi 3 Model B utilizado no projeto.

Figura 9: *Raspberry Pi 3 Model B*



Fonte: ([RASPBERRYPI, 2019](#)).

O uso da Raspberry Pi se fez necessário no projeto pois foi decidido utilizar um servidor local para funcionar como Broker MQTT e guardar as informações no banco de dados coletadas durante os experimentos. Como a placa é capaz de rodar um sistema operacional Linux, foi fácil de instalar e desenvolver os componentes necessários para atenderem os requisitos do projeto. Na tabela 4 é possível ler as especificações técnicas do módulo:

Tabela 4: Esepcificações técnicas da *Raspberry Pi 3 Model B*

Arquitetura	Cortex-A53 (ARMv8) de 64 bits
Processador	Broadcom BCM2837B0 Quad Core de 1.4 GHz
Memória SDRAM	1 GB LPDDR2 SDRAM
USB	4 portas USB 2.0
Entrada de Vídeo	Conector CSI de 15 pinos
Saída de Vídeo	Porta HDMI 1.3
Saída de Áudio	Saída estéreo de 4 pólos
Armazenamento	Suporte para cartão SD
Conectividade sem fio	WiFi 802.11, Bluetooth 4.2, BLE
Rede com fio	Ethernet USB 2.0 (velocidade máxima de 300 Mbps)
Consumo	700 mA (3.5 W)
Alimentação	5V via Micro USB, suporte a PoE

Fonte: ([RASPBERRYPI, 2019](#)).

2.10 Sistema de gerenciamento dos dados

2.10.1 Servidor Apache

O servidor Apache é um projeto de servidor web *open-source* criado em 1995 por Rob McCool. Segundo ([NETCRAFT, 2019](#)), em maio de 2019 o Apache é o servidor mais utilizado nos sites ativos do mundo, com 29,85% de todo o mercado com um total de 56 milhões de sites. Uma das vantagens de ser um código aberto é que seus usuários podem alterar o código fonte a fim de adequar às suas necessidades. A principal razão que o servidor Apache foi escolhido para esse projeto foi sua facilidade de instalação e, depois de combinado com o software phpMyAdmin se torna fácil o gerenciamento de um banco de dados para um servidor local. A versão utilizada no projeto foi a 2.4, instalado em ambiente Ubuntu 16.04 operando na *Raspberry Pi*.

2.10.2 MySQL

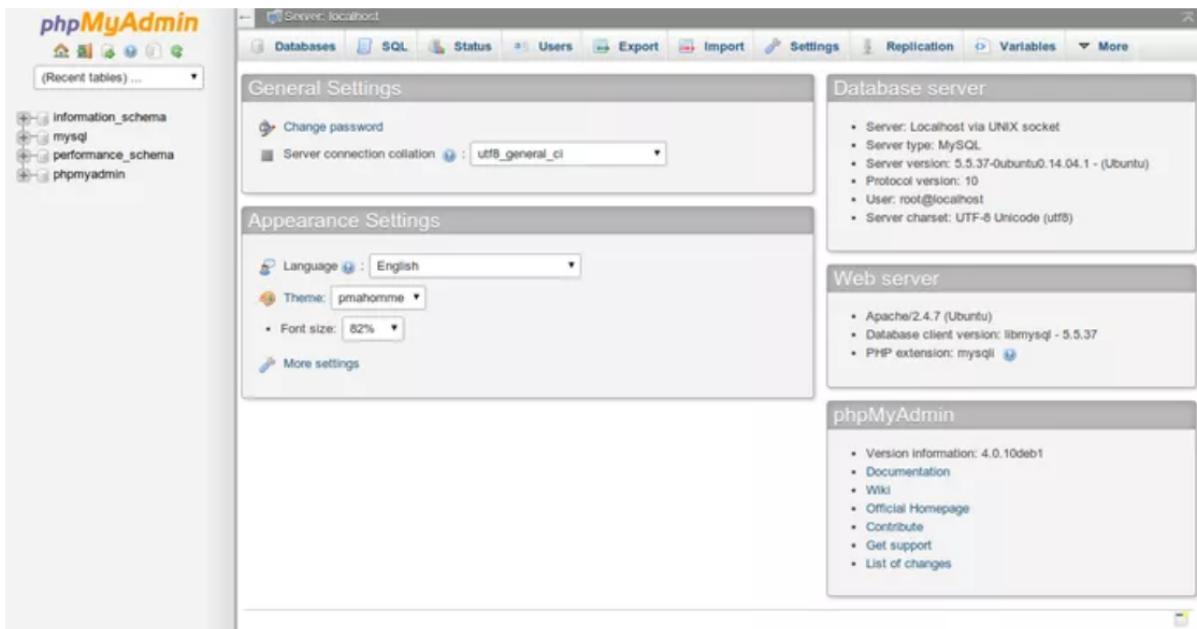
O MySQL é um Sistema de Gerenciamento de Banco de Dados (SGBD), que utiliza a linguagem *Structured Query Language* (SQL) como interface. O sistema se tornou o mais popular banco de dados *open-source* do mundo porque possui consistência, alta performance, confiabilidade e é fácil de usar. Além disso, o MySQL se tornou a escolha de uma nova geração de aplicações, assim como esse projeto, que utilizam o modelo LAMP (Linux, Apache, MySQL, PHP). Para o projeto foi instalado a versão 5.7 do servidor MySQL, em ambiente Ubuntu 16.04 instalado na *Raspberry Pi*.

2.10.3 phpMyAdmin

A ferramenta phpMyAdmin é um software *open-source* desenvolvido em PHP com o intuito de lidar com a administração do MySQL pela Internet ([PHPMYADMIN, 2019](#)). É

possível criar base de dados, adicionar, excluir e editar tabelas, adicionar,remover e editar campos, executar comandos SQL além de manipular os campos chaves. A ferramente é muito utilizada por programadores web que necessitam gerenciar as bases de dados de forma prática e intuitiva. A figura 10 ilustra a interface principal do programa.

Figura 10: Tela inicial do phpMyAdmin



Fonte: Elaborado pelo autor..

3 DESENVOLVIMENTO

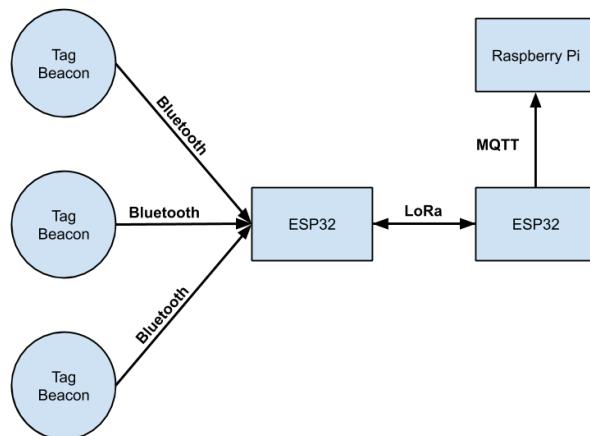
3.1 Projeto

Esse capítulo apresenta em detalhes como o sistema todo foi projetado, isso inclui os módulos *Heltec WiFi LoRa 32* que foram configurados para atuar de estação coletora e estação central, e as implementações na *Raspberry Pi* para atuar tanto como Broker MQTT com o software Mosquitto quanto Cliente MQTT no Node-RED para enviar os dados no Banco de Dados local no servidor Apache.

Antes de entrar nos detalhes de cada componente do sistema, será apresentado uma visão geral do projeto como um todo. Para ser possível a localização dos animais na fazenda, é necessário que cada animal possua um *beacon* para identificação individual, isso significa que cada *beacon* possui um único UUID e está preso fisicamente ao animal, sua função é enviar pacotes de dados para a estação coletora via BLE. A estação coletora recebe esses dados e tenta enviar via *LoRa* para a estação central, caso receba uma mensagem de confirmação ela volta escanear ou tentar enviar o próximo pacote. A estação central recebe os dados e repassa para a *Raspberry Pi* via MQTT, que processa esses dados e finalmente envia para o banco de dados. A figura 11 ilustra o sistema.

Todos os códigos desenvolvidos durante o projeto estão disponíveis em <<https://github.com/LucasSossai/tcc-animaltracker-esp32>>. As pastas "EstacaoCentral" e "EstacaoColetora" contém os códigos que foram utilizados nos módulos *Heltec WiFi Lora 32* e o arquivo "FlowNodeRed" se refere ao *flow* utilizado no Node-RED.

Figura 11: Diagrama do sistema



Fonte: Elaborada pelo autor.

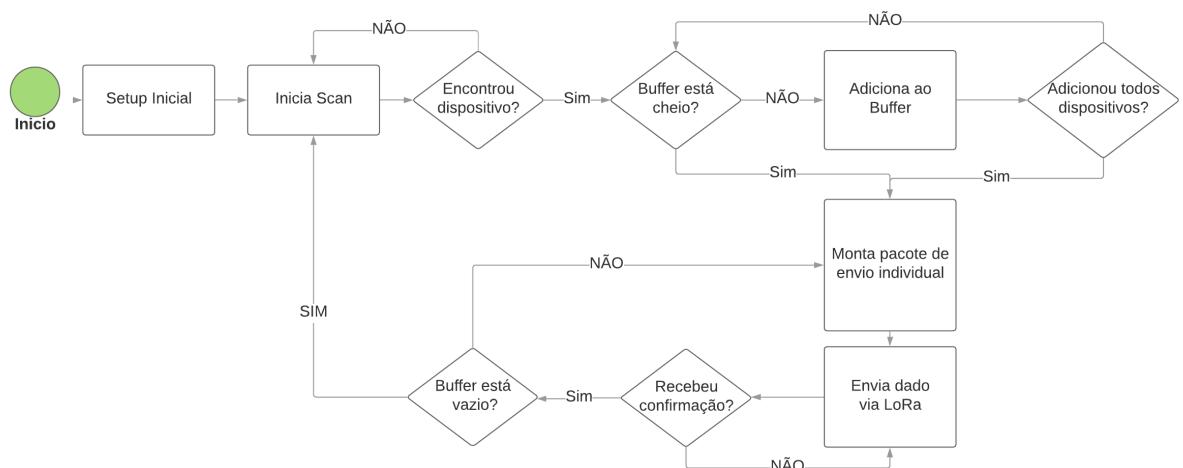
3.2 Desenvolvimento dos módulos *Heltec WiFi LoRa 32*

3.2.1 Estação coletora

Como o próprio módulo já possui todos componentes necessários para o sistema como comunicação BLE e *LoRa*, para a montagem só foi necessário conectar a placa via cabo USB para comunicação serial com a IDE *Arduino* e desenvolver o software.

O software é responsável por configurar o ESP32 para escanear os *beacons* próximos, processar os dados para extrair a UUID e a RSSI, armazenar em um *buffer* e enviar via *LoRa* assim que cada ciclo de *scan* acaba. Depois de enviar um pacote contendo informações referentes à um *beacon*, o programa aguarda receber um sinal de resposta da estação central, caso não receba ele tenta enviar novamente o pacote até obter sucesso e enviar o próximo dado. Quando todos os dados referentes à um *scan* são enviados, o programa inicia o processo de busca de novos *beacons* novamente. A figura 12 ilustra o fluxograma do funcionamento da estação coletora.

Figura 12: Fluxograma do software da estação coletora.

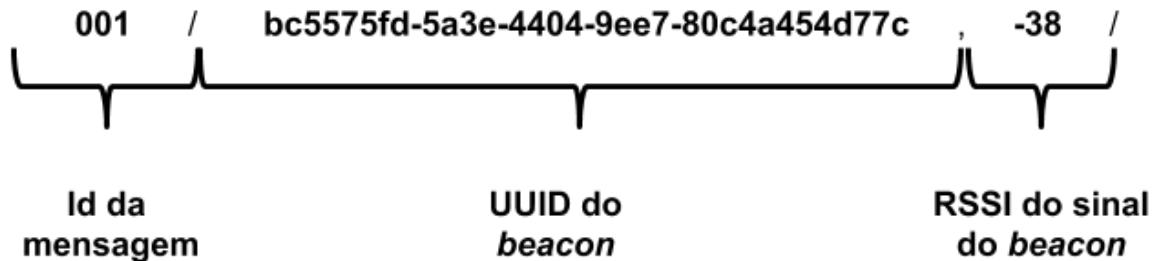


Fonte: Elaborada pelo autor.

3.2.2 Comunicação entre os módulos

Para garantir que os dados da estação coletora cheguem à estação central, a comunicação entre elas precisa ser bidirecional, para isso o pacote de dados foi estruturado para ser conforme a figura 13. A Id da mensagem é um valor inteiro que inicia em 0 e incrementa toda vez que um pacote é enviado com sucesso. A estação coletora considera que uma mensagem foi enviada com sucesso quando a estação central responde a Id da mensagem de volta em menos de 500 ms. Para garantir que a informação recebida está consistente com a enviada, o software utiliza o método CRC.

Figura 13: Exemplo de pacote de dados enviado via *LoRa*

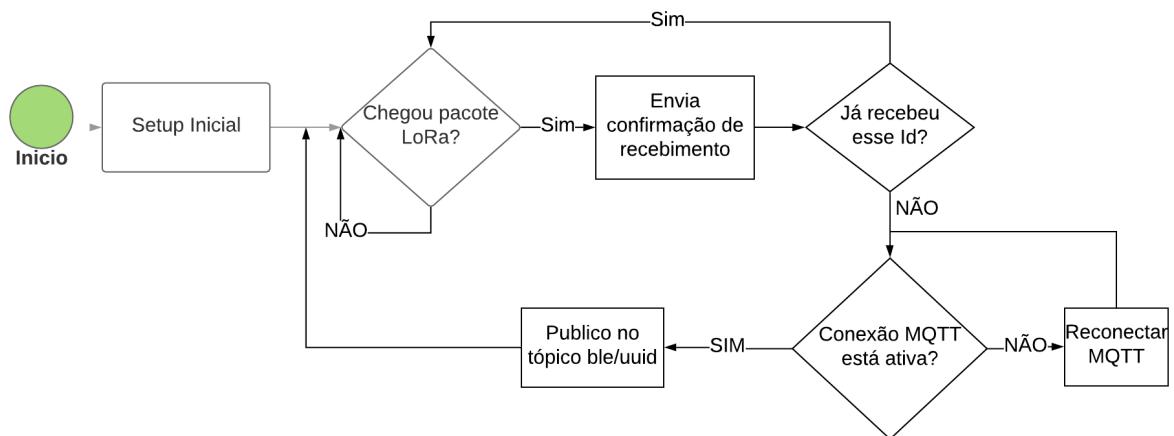


Fonte: Elaborada pelo autor.

3.2.3 Estação central

O comportamento do software da estação central está ilustrado na figura 14. Depois de iniciado, o módulo tenta conectar no *WiFi* local e no broker MQTT criado pela *Raspberry Pi*, se obtém sucesso ele começa a aguardar o recebimento de sinais *LoRa*. Sempre que recebe um pacote de dados, o módulo envia a confirmação, checa se a Id desse pacote já foi enviada via MQTT e caso não foi ainda, publica o dado recebido no tópico "ble/uuid".

Figura 14: Fluxograma do software da estação central.



Fonte: Elaborada pelo autor.

3.3 Desenvolvimento da *Raspberry Pi*

3.3.1 Mosquitto

Para implementar a estrutura MQTT foi utilizado o *broker* Mosquitto. Para construir uma rede local WiFi baseada em MQTT primeiramente foi configurado um roteador como ponto de acesso de uma rede local sem fio e depois conectamos a *Raspberry Pi* à rede. Em seguida, foi instalado o servidor Mosquitto MQTT e a porta foi mantida no valor *default* de 1883. Depois de instalado, o *broker* inicia automaticamente no IP que a *Raspberry Pi* se encontra e qualquer dispositivo conectado à rede WiFi e com o endereço do broker já pode se subscrever e publicar nos tópicos.

3.3.2 Node-RED

O *flow* desenvolvido na ferramenta Node-RED, ilustrado na figura 15, consiste de 3 nós:

- **Cliente MQTT:** Responsável de se conectar ao Mosquitto e subscrever nos tópicos. Para a conexão, é necessário informar o IP e a porta do *broker* além de quais tópicos quer subscrever.
- **Parser:** Função responsável de receber os dados publicados no tópico "ble/uuid", processar esses dados para extrair a UUID e RSSI de cada pacote e montar o comando SQL para inserir as informações no banco de dados Apache. O código em JavaScript da função está abaixo da figura 15.
- **Cliente Banco de Dados:** Nó responsável de se conectar ao servidor Apache e processar os comandos SQL.

Figura 15: Flow principal do Node-RED.



Fonte: Elaborada pelo autor.

Função *Parser*:

```

var responseList = [];
var auxList = msg.payload.split('/');
msg.topic="INSERT INTO blescan (uuid, rssи) VALUES ";
auxList.splice(0,1);
  
```

```

if(auxList.length>0&&auxList[0]!="x" && auxList[0]!==',' &&
msg.topic=="ble/uuid"){
    auxList.forEach(function(item){
        var list = item.split(',');
        if(list[0]==','&& list[1]==','){
            responseList.push(list[0]);
        if(!list[1]){
            responseList.push(0);
        }else{
            responseList.push(list[1]);
        }
        msg.topic+=(?,?),";
    })
    msg.topic=msg.topic.substring(0,msg.topic.length-1);
    msg.payload=responseList;
    return msg;
}

```

3.3.3 Banco de Dados no servidor Apache

Depois de instalado o Servidor Apache e software phpMyAdmin na *Raspberry Pi*, foi possível criar a tabela nomeada *blescan* ilustrada na figura 16. Os 4 atributos da tabela são descritos abaixo:

- **uuid:** Entrada no formato de texto com identificação única do beacon
- **rssi:** Valor inteiro para inserir a potência do sinal recebido
- **timestamp:** Preenchido automaticamente sempre que a uuid e rssi são inseridos com o valor de tempo atual
- **ID:** *Primary-Key* da tabela, diferente da ID transmitida via *LoRa*, um número inteiro que auto incrementa sempre que são inserido valores na tabela

Figura 16: Estrutura da tabela blescan no servidor Apache.

Nome	Tipo	Colação	Atributos	Nulo	Padrão	Extra
uuid	text	latin1_swedish_ci		Não	Nenhum wrap (padrão: none)	
rssi	int(11)			Não	Nenhum wrap (padrão: none)	
timestamp	timestamp		on update CURRENT_TIMESTAMP	Não	CURRENT_TIMESTAMP	ON UPDATE CURRENT_TIMESTAMP
ID 🔑	int(11)			Não	Nenhum wrap (padrão: none)	AUTO_INCREMENT

Fonte: Elaborada pelo autor.

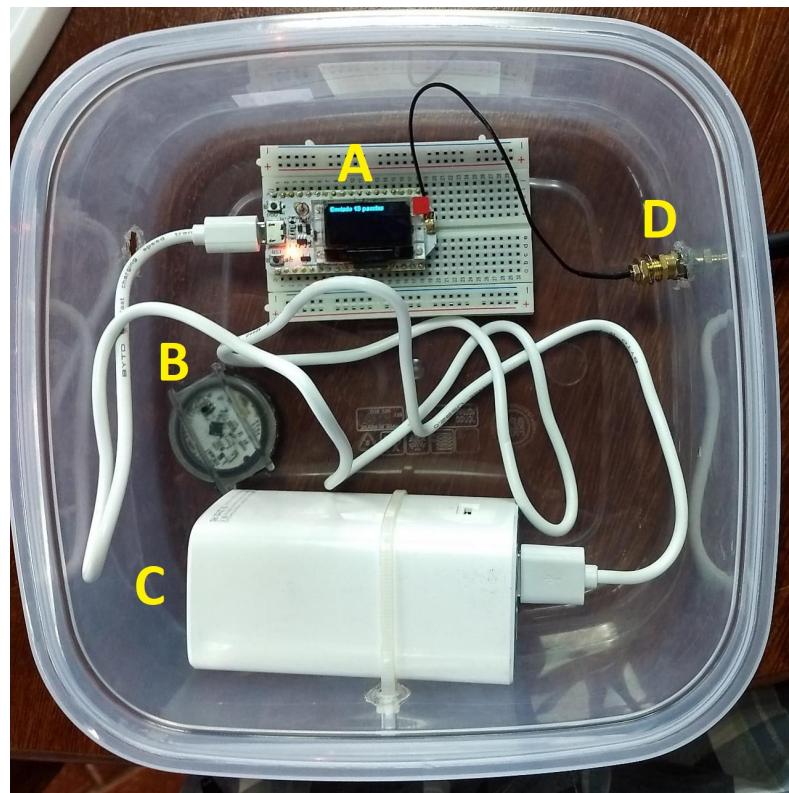
4 EXPERIMENTOS E RESULTADOS

Durante o desenvolvimento do projeto foram realizados 2 experimentos, o primeiro foi feito para descobrir qual a taxa de perda de pacotes LoRa enquanto o segundo experimento avaliou a velocidade de transmissão de pacotes. Ambos experimentos foram feitos com a estação central posicionada na sede da fazenda com acesso a internet e a estação coletora variando de posição para avaliar diferentes cenários. Um *beacon* ficou armazenado dentro da estação coletora durante os experimentos para garantir a presença de um sinal *bluetooth* e poder avaliar melhor a transmissão LoRa.

4.1 Estação coletora

Na figura 17 é possível observar a estação coletora montada para realizar os testes. Na imagem temos um módulo *Heltec WiFi LoRa 32* (A) com sua antena (D) para comunicação *LoRa*, um *beacon* (B) e uma bateria (C). Para a montagem da estação, foi utilizado uma caixa de plástico onde uma *protoboard* foi fixada junto do módulo principal, utilizando um ferro quente foi aberto um buraco para saída da antena (para evitar qualquer tipo de interferência no sinal) e dois buracos para fixar a bateria com uma abraçadeira.

Figura 17: Estação coletora utilizada nos testes.



Fonte: Elaborado pelo autor.

4.2 Ambiente de testes

Para realizar os testes e demonstrar o sistema proposto, foi escolhido um cenário rural onde foi possível testar a velocidade e o alcance da transmissão dos pacotes entre os módulos *Heltec WiFi LoRa 32*. No primeiro experimento, a estação coletora foi posicionada em 11 pontos diferentes enquanto no experimento 2 foi possível replicar em apenas 7 pontos pois 4 pontos estão no meio da represa e no dia do segundo experimento não foi possível sair de barco para coleta de dados. Durante todos experimentos um único *beacon* ficou emitindo sinal durante aproximadamente uma hora em cada posição, enquanto o módulo com acesso à rede local foi fixado na sede da fazenda enviando os dados para a *Raspberry Pi*, que por sua vez armazena as informações no banco de dados. Na figura 18 é possível observar o mapa, criado através da ferramenta Google Earth onde foram realizados os testes. As marcações amarelas correspondem às coordenadas dos pontos de teste, obtidas através do sinal GPS de um celular durante o experimento.

Figura 18: Mapa elaborado no Google Earth com os pontos onde foram realizados os testes.

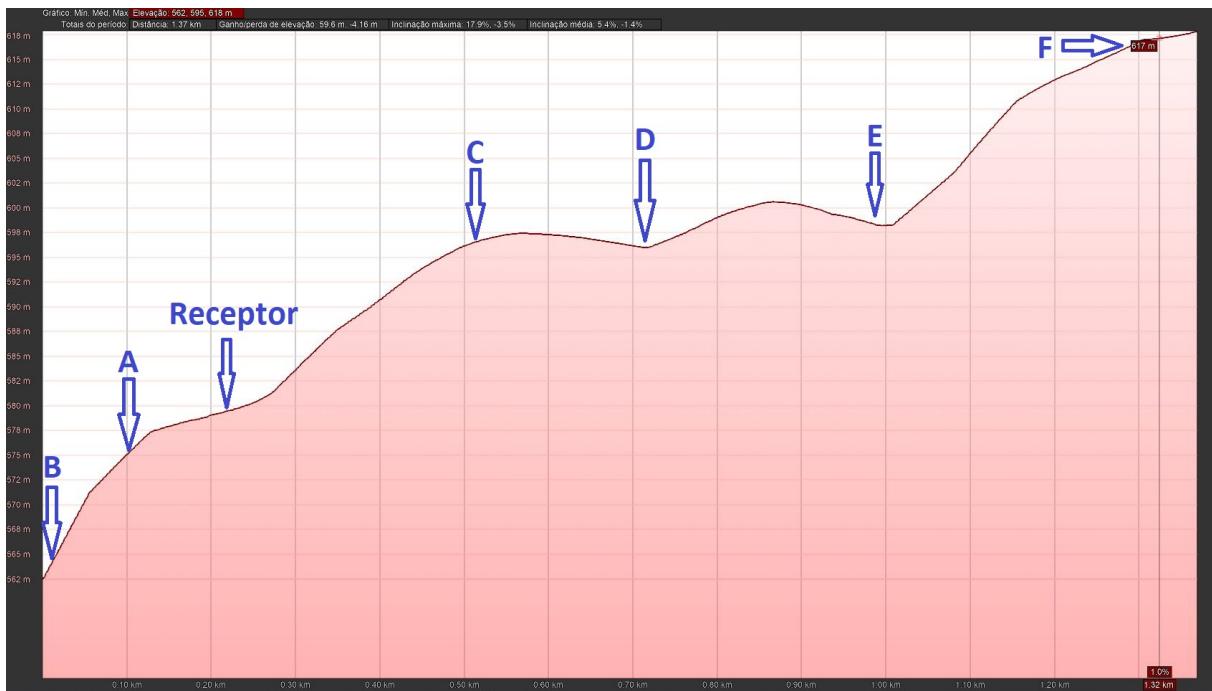


Fonte: Elaborado pelo autor.

Por meio da ferramenta Google Earth foi possível gerar o perfil de elevação onde foram realizados os testes. Para gerar o gráfico ilustrado na figura 19 foi traçado uma linha da beira da água próximo ao ponto B e no nível dos pontos G,H,I e J, até o Ponto F, na entrada da fazenda. Os pontos A,C,H e I possuíam linha de visão direta limpa entre a estação coletora e a estação central enquanto no meio dos pontos B,D,E,F,G e J e o receptor haviam obstáculos como árvores e relevo. A maior distância entre os pontos foi de 2131m quando o a estação coletora estava no ponto J. Os pontos G,H,I e J estão no

mesmo nível de altura posicionados no meio da Represa Jurumirim na cidade de Avaré, SP, tais pontos foram escolhidos por possuir na maioria dos casos linha de visão direta sem obstáculos e possibilidade de medir em maiores distâncias sem interferência de relevo. Nesse caso, foi necessário o uso de um barco para transporte da estação coletora até os locais de teste.

Figura 19: Perfil de elevação do ambiente de testes.



Fonte: Elaborado pelo autor.

4.3 Experimento 1: Avaliação da taxa de perda de pacotes

O primeiro experimento foi realizado para avaliar a taxa de perda de pacotes via comunicação LoRa. Para isso dois contadores foram implementados, o contador da estação coletora incrementa sempre que tenta enviar um pacote LoRa, com tamanho fixo de 10 bytes, enquanto o contador da estação central aumenta sempre que recebe esse pacote de dados.

4.3.1 Resultados

A tabela 5 apresenta os resultados obtidos no primeiro experimento. Os pontos A e C mantiveram uma taxa próxima da referência enquanto os pontos D e E tiveram a maior taxa de perda comparado aos outros. Comparando os valores da tabela com as figuras 18 e 19 é possível observar que, quando a estação coletora está posicionada em um vale sem visão direta com a estação central, a taxa de perda é maior do que quando estamos a uma distância maior mas sem obstáculos.

Tabela 5: Tabela de resultados da taxa de perda de pacotes.

Local	Distância entre Tx e Rx (m)	Diferença de altitude entre Tx e Rx (m)	Tentativas de envio da estação coletora	Total recebido pela estação central	Taxa de Perda (%)
Referência	0,1	0	1129	1116	1,15 %
Ponto A	115	-4	1002	972	2,99 %
Ponto B	211	-15	939	735	21,73 %
Ponto C	318	17	936	879	6,09 %
Ponto D	557	18	1060	580	45,28 %
Ponto E	777	21	991	363	63,37 %
Ponto F	1087	38	982	764	22,20 %
Ponto G	473	-19	981	732	25,38 %
Ponto H	1023	-19	1023	873	14,66 %
Ponto I	2069	-19	931	707	24,06 %
Ponto J	2131	-19	935	593	36,58 %

4.4 Experimento 2: Avaliação da velocidade de transmissão de pacotes

O segundo experimento foi realizado utilizando o método desenvolvido em 3.2.2, com a estação coletora enviando pacotes de 45 bytes e a central respondendo com 3 bytes. Para proteção extra contra a chuva, a caixa também foi embalada por sacolas plásticas para evitar que água entrasse dentro dela. Na figura 20 temos a estação coletora em funcionamento no ponto D em um dia chuvoso. A estação colheu dados por mais de 6 horas sem que fosse notado a entrada de qualquer umidade na caixa.

Figura 20: Estação coletora enviando dados no Ponto D.



Fonte: Elaborado pelo autor.

4.4.1 Resultado

Os resultados obtidos são mostrados na tabela 6:

Tabela 6: Características e resultados dos pontos de teste

Local	Distância entre Tx e Rx (m)	Diferença de altitude entre Tx e Rx (m)	Velocidade de transmissão (pacotes/hr)
Referência	0,1	0	839
Ponto A	115	-4	827
Ponto B	211	-15	693
Ponto C	318	17	812
Ponto D	557	18	44
Ponto E	777	21	23
Ponto F	1087	38	116

Fonte: Elaborado pelo autor.

Comparando a figura 19 com a tabela 6 é possível concluir que o principal desafio do sistema é transmitir quando há relevo pois, para distâncias de até 300m a velocidade de transmissão se manteve próxima da referência, menos no Ponto B quando não havia linha de visão direta entre transmissor e receptor. Também é possível observar que houve uma queda brusca na velocidade de transmissão nos Pontos D e E e que a velocidade aumentou ao chegar no Ponto F, ou seja, mesmo com uma distância maior, no Ponto F foi possível obter um melhor resultado pois havia menos obstáculos até o receptor do que nos pontos D e E.

5 CONCLUSÃO

5.1 Resumo

O sistema desenvolvido nesse projeto pode contribuir para o desenvolvimento tecnológico nas fazendas, permitindo um melhor monitoramento dos animais e provavelmente facilitando a tomada de decisão dos profissionais da área. Os resultados provaram a hipótese de que é possível transmitir os dados a uma distância de até 1 Km via LoRa, possibilitando a coleta dos dados das estações coletoras até a sede da fazenda. No entanto vale ressaltar que quando não há linha direta de visão entre a estação coletora e estação central a taxa de perda de pacotes aumenta significativamente como é o apresentado nas tabelas 5 e 6 no Ponto B, que tem menor velocidade que o Ponto C mesmo com menor distância horizontal, e nos Pontos D e E, que como estão posicionados em um vale, o relevo atrapalha a transmissão. Esse trabalho também pode ser utilizado como base para estudos, projetos e aplicações futuras, além de também ser possível a extensão para outras aplicações, devido a grande quantidade de uso ainda não explorados por parte dos *beacons*.

5.2 Vantagens e Desvantagens

O sistema proposto tem a vantagem dos *beacons* de possuir baixo consumo energético, levando à uma maior autonomia, diferente dos sistemas convencionais que usam 3G e que dependem de conexão com sinais de operadoras, ainda de baixa disponibilidade em zonas rurais. As desvantagens são que a distância máxima de comunicação entre a estação coletora e a estação central depende da topografia de cada terreno, além de que é necessário ter estações coletoras em toda a área monitorada, aumentando o custo do sistema para terrenos muito extensos.

5.3 Relacionamento entre o Curso e o Projeto

O curso de Engenharia Elétrica com Ênfase em Eletrônica apresenta uma grade curricular ampla, possibilitando ao aluno o aprendizado em diversas áreas ligadas à eletrônica. Durante o desenvolvimento do projeto, disciplinas relacionadas à programação, lógica e telecomunicação forneceram a base para realização do trabalho.

5.4 Trabalhos Futuros

Pode-se listar as seguintes sugestões para trabalhos futuros:

- **Triangulação do sinal:** É possível estimar a posição do animal em um ambiente conhecido à partir de 3 receptores *beacon* sem a necessidade do uso de GPS, podendo

assim alertar os proprietários dos animais sobre eventuais fugas.

- **Eletrônica para geração de energia solar:** Dimensionar painéis solares, bateria e controlador de carga para que a estação coletora possa operar de forma autônoma.
- **Interface Web** Com o sistema de coleta de dados pronto, pode-se desenvolver uma interface web para apresentar informações relevantes aos donos dos animais.
- **Processamento dos dados na Cloud:** Caso a estação central tenha acesso à internet na sede, pode-se redirecionar os dados diretamente para um serviço Cloud, como por exemplo a AWS. Com isso, não é mais necessário utilizar a *Raspberry Pi* como servidor local.
- **Antena própria:** Como a estação central é fixa, é possível projetar uma antena direcional à fim de obter melhores resultados na comunicação LoRa.

REFERÊNCIAS

- ABIEC. **Beef Report**: Os principais dados que mostram o perfil da pecuária brasileira. [s.n.], 2018. Disponível em: <<http://abiec.com.br/Sumario2019.aspx>>. Acesso em: 17 abril 2019.
- ALBUQUERQUE, M. M. **Rede de sensores para aplicação em agricultura: um estudo de caso**. 2009. Dissertação (Mestrado), Porto Alegre, 2009.
- ARDUINO, S. A. Arduino. **Arduino LLC**, 2015.
- ATZORI, L. The internet of things: a survey. **Computer Networks**, 2010.
- AYELE, E. D. Performance analysis of lora radio for an indoor iot applications. In: . [S.l.: s.n.], 2017.
- BLACKSTOCK, M.; LEA, R. Toward a distributed data flow platform for the web of things (distributed node-red). In: ACM. **Proceedings of the 5th International Workshop on Web of Things**. [S.l.], 2014. p. 34–39.
- CAMARGO, J. F. C. **Desenvolvimento de tecnologia de hardware e software para o monitoramento de animais**. 2018. Dissertação (Mestrado), São Carlos, 2018.
- DEUGO, D. Using beacons for attendance tracking. In: THE STEERING COMMITTEE OF THE WORLD CONGRESS IN COMPUTER SCIENCE, COMPUTER **Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)**. [S.l.], 2016. p. 155.
- FONSECA, E.; VEGA, A. de la. Tutorial sobre introduçao a projetos utilizando o kit de desenvolvimento arduino (anais de congresso). In: **XXXIX Congresso Brasileiro de Educação em Engenharia**. [S.l.: s.n.], 2011.
- GOMEZ, C.; BOSCH, J. O.; PARADELLS, J. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. **Sensors (Basel, Switzerland)**, v. 12, p. 11734–53, 12 2012.
- GOULART, V. Sistema iot para monitoramento de porteiros utilizando lora e lorawan. **CIMATech**, v. 1, n. 5, 2018.
- HELTEC. **Página do produto Heltec Wifi Lora 32**. [s.n.], 2018. Disponível em: <<https://heltec.org/project/wifi-lora-32>>. Acesso em: 14 de maio 2019.
- LAMPKIN, V. et al. **Building smarter planet solutions with mqtt and ibm websphere mq telemetry**. [S.l.]: IBM Redbooks, 2012.
- LOCKE, D. Mq telemetry transport (mqtt) v3. 1 protocol specification. **IBM developerWorks Technical Library**, p. 15, 2010.
- MELO, E. C. A. de et al. Uma arquitetura de beacons customizáveis para internet das coisas. In: **9º Simposio Brasileiro de Computacao Ubiqua e Pervasiva (SBCUP 2017)**. Porto Alegre, RS, Brasil: SBC, 2017. v. 9. ISSN 2595-6183. Disponível em: <<https://portaldeconteudo.sbc.org.br/index.php/sbcup/article/view/3307>>.

-
- NETCRAFT. **May 2019 Web Server Survey**. 2019. Disponível em: <<https://news.netcraft.com/archives/2019/05/10/may-2019-web-server-survey.html>>.
- PHPMYADMIN. **Bringing MySQL to the web**. 2019. Disponível em: <<https://www.phpmyadmin.net/>>.
- RASPBERRYPI. **Página do produto Raspberry Pi 3 Model B**. [s.n.], 2019. Disponível em: <<https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>>. Acesso em: 14 de maio 2019.
- REYNDERS, B.; MEERT, W.; POLLIN, S. Range and coexistence analysis of long range unlicensed communication. In: . [S.l.: s.n.], 2016. p. 1–6.
- REYNDERS, B.; POLLIN, S. Chirp spread spectrum as a modulation technique for long range communication. In: IEEE. **2016 Symposium on Communications and Vehicular Technologies (SCVT)**. [S.I.], 2016. p. 1–5.
- SEMTECH. **What is LoRa®?** 2019. Disponível em: <<https://www.semtech.com/lora>>.
- SISINNI, E. et al. Delay estimation of industrial iot applications based on messaging protocols. **IEEE Transactions on Instrumentation and Measurement**, v. 67, 04 2018.
- ZAFARI, F.; PAPAPANAGIOTOU, I. Enhancing ibeacon based micro-location with particle filtering. In: . [S.l.: s.n.], 2015. p. 1–7.