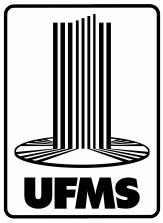
Willyan Candido Silva

Barramento de Comunicação IoT para Plataforma de Pecuária de Precisão



UNIVERSIDADE FEDERAL
DE MATO GROSSO DO SUL

Campo Grande – MS

Junho de 2021

Willyan Candido Silva

Barramento de Comunicação IoT para Plataforma de Pecuária de Precisão

Dissertação de mestrado apresentada ao Programa de Mestrado Stricto Sensu em Computação Aplicada, da Faculdade de Computação, mantido pela Fundação Universidade Federal de Mato Grosso do Sul, como parte dos requisitos para a obtenção do título de Mestre em Computação Aplicada (Área de Concentração: Aplicações Distribuídas).

Universidade Federal de Mato Grosso do Sul – UFMS

Faculdade de Computação – FACOM

Programa de Pós-Graduação em Computação Aplicada

Orientador: Profa. Dra. Hana Karina Salles Rubinsztejn

Coorientador: Dr. Camilo Carromeu

Campo Grande – MS

Junho de 2021

Willyan Candido Silva
Barramento de Comunicação IoT para Plataforma de Pecuária de Precisão/ Willyan Candido Silva. – Campo Grande – MS, Junho de 2021-
73p. : il. (algumas color.) ; 30 cm.

Orientador: Profa. Dra. Hana Karina Salles Rubinsztejn
Coorientador: Dr. Camilo Carromeu

Dissertação (Mestrado) – Universidade Federal de Mato Grosso do Sul – UFMS
Faculdade de Computação – FACOM
Programa de Pós-Graduação em Computação Aplicada, Junho de 2021.

1. Pecuária de Precisão. 2. Internet das Coisas. 2. Automação. I. Hana Karina Salles Rubinsztejn. II. Universidade Federal de Mato Grosso do Sul. III. Faculdade de Computação. IV. Barramento de Comunicação IoT para Plataforma de Pecuária de Precisão.

Willyan Candido Silva

Barramento de Comunicação IoT para Plataforma de Pecuária de Precisão

Dissertação de mestrado apresentada ao Programa de Mestrado Stricto Sensu em Computação Aplicada, da Faculdade de Computação, mantido pela Fundação Universidade Federal de Mato Grosso do Sul, como parte dos requisitos para a obtenção do título de Mestre em Computação Aplicada (Área de Concentração: Aplicações Distribuídas).

Profa. Dra. Hana Karina Salles Rubinsztejn
Orientador

Dr. Camilo Carromeu
Coorientador

Prof. Dr. Fábio Iaione
Membro Interno

Prof. Dr. Luciano Gonda
Membro Externo

Campo Grande – MS
Junho de 2021

Resumo

A pecuária brasileira vem demonstrando ao longo dos anos sua primazia no mercado mundial de carne bovina. Em decorrência dessa conjuntura e sua importância para o setor econômico do país, desenvolver e aprimorar os meios empregados na cadeia de criação de gado de corte é crucial para aumentar a produção, reduzir os custos e assegurar o constante progresso exigido pelo mercado. Nesse contexto, a utilização de uma infraestrutura pautada na Internet das Coisas (*Internet of Things* – IoT) contribui para um monitoramento de baixo custo e confere a oportunidade de automatizar as propriedades produtoras de carne bovina. A FACOM/UFMS e a Embrapa Gado de Corte possuem várias pesquisas e projetos voltados para a pecuária de precisão, dentre eles a plataforma e-Cattle. O objetivo desse trabalho é fornecer um barramento de comunicação com protocolos IoT para integração de uma maior variedade de sensores e a persistência das informações coletadas na plataforma. Dessa forma, o uso de protocolos de comunicação IoT, como por exemplo LoRA e MQTT, aumentarão a heterogeneidade de sensores aceitos, contribuindo para uma solução de baixo custo de implantação e monitoramento.

Palavras-chave: Pecuária de Precisão. Automação. Internet das Coisas. Sensores.

Abstract

Brazilian cattle breeding has demonstrated over the years its primacy in the world beef market. As a result of this situation and its importance for the country's economic sector, developing and improving the means used in the beef cattle production chain is crucial to increase production, reduce costs and ensure the constant progress required by the market. In this context, the use of an infrastructure based on the Internet of Things (IoT) contributes to low-cost monitoring and provides the opportunity to automate beef-producing properties. FACOM / UFMS and Embrapa Gado de Corte have several researches and projects aimed at precision cattle breeding, among them the e-Cattle platform. The objective of this work is to provide a communication bus with IoT protocols for the integration of a greater variety of sensors and the persistence of information collected on the platform. Thus, the use of IoT communication protocols, as for example LoRA and MQTT, will increase the heterogeneity of accepted sensors, contributing to a low-cost implementation and monitoring solution.

Keywords: Precision Livestock. Automation. Internet of Things. Sensors.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ABIEC	Associação Brasileira das Indústrias Exportadoras de Carnes
API	*Application Programming Interface*
BLE	*Bluetooth Low Energy*
EMBRAPA	Empresa Brasileira de Pesquisa Agropecuária
FACOM	Faculdade de Computação
FOTA	*Firmware Over-The-Air*
HTTP	*Hypertext Transfer Protocol*
IA	Inteligência Artificial
IoT	*Internet of Things*
LoRa	*Long Range*
LPWAN	*Low Power Wide Area Network*
M2M	*Machine to Machine*
MQTT	*Message Queuing Telemetry Transport*
QoS	*Quality of Service*
RFID	*Radio-Frequency IDentication*
RSSF	Redes de Sensores Sem Fio
TI	Tecnologia da Informação
TIC	Tecnologias da Informação e Comunicação
TTN	*The Things Network*
UFMS	Universidade Federal de Mato Grosso do Sul

Sumário

1 Introdução

Responsável por $8,5\%$ do Produto Interno Bruto (PIB), equivalente à R$7,3 trilhões no ano de 2019, a pecuária de corte é um fator importante na balança comercial brasileira. Desse montante, o PIB da pecuária de corte representa R$618,5 bilhões e o saldo do agronegócio vem apresentando resultados positivos a mais de três décadas (ABIEC, 2020), demonstrando que é um setor consistente, sólido e de grande importância para o Brasil.

O valor movimentado pela pecuária engloba uma cadeia de transações que vai além da venda de animais. Dentre os negócios que compõem essa categoria estão: insumos utilizados na pecuária, investimentos em genética, sanidade animal, nutrição, entre outros (ABIEC, 2020).

O Brasil possui a maior rebanho de bovinos no mundo, um total de 213,7 milhões de cabeças. Contudo, fica em segundo lugar em produção com $14,8\%$, atrás apenas dos Estados Unidos com $17,3\%$. Analisando o contexto nacional, o estado de Mato Grosso do Sul (MS), é responsável por $9,82\%$ do rebanho nacional e ocupa a quarta posição no *ranking* de produtores (ABIEC, 2020).

Na atualidade, vivenciamos um momento de extrema necessidade de inovação e do emprego de tecnologias para poder manter o crescimento econômico e de produtividade. A capacidade de inovar é uma ferramenta estratégica para as empresas que desejam manter sua posição competitiva no mercado, e o setor do agronegócio não é diferente.

O processo de digitalização tem alterado a maneira como o agronegócio tem conduzido seu processo de criação, administração e negócio. Hoje, as empresas estão enfrentando algo novo com a adoção de tecnologias disruptivas e acessíveis, como Computação Móvel, Internet das Coisas (*Internet of Things* – IoT), Nuvem, Nanotecnologia, Inteligência Artificial (IA) e *Machine Learning*. Inevitavelmente a digitalização é um processo de transformação pelo qual as empresas devem atravessar para se manterem no mercado e usufruir dos benefícios que essas tecnologias criam (BUCCI; BENTIVOGLIO; FINCO, 2018).

O uso de tecnologias modernas como sensores, robótica, análise de dados, sistemas embarcados fazem parte da modernização do campo, o uso dessas tecnologias nas diversas áreas de processo de produção proporciona economia, agregação de valor à produção e promove uma tomada de decisão mais ágil ao produtor (SHAMSHIRI et al., 2018). Difundida na sociedade e presente em muitas empresas que não são tradicionalmente do setor de Tecnologia da Informação e Comunicação (TIC), essas tecnologias são responsáveis pelo crescimento da produção e da receita (MILLER; ATKINSON, 2014).

Toda essa revolução na cadeia produtiva teve início no final da década de 90 com o pesquisador do Instituto de Tecnologia de Massachusetts (*Massachusetts Institute of Technology* - MIT) chamado Ashton (2009), que utilizou o termo IoT em sua palestra para empresa Procter & Gamble (P&G) com objetivo de chamar atenção dos executivos. Sua ideia era de que os produtos produzidos por eles poderiam ser etiquetados eletronicamente com identificadores de radiofrequência (*Radio-Frequency IDentification* – RFID) facilitando a logística na linha de produção.

Evans (2011) diz que a quantidade de dispositivos conectados superou a população mundial antes do ano de 2010. A quantidade de dispositivos que se conectavam à Internet em 2003 era em torno de 500 milhões, enquanto que a população mundial aproximadamente $6,3$ bilhões. A razão entre dispositivos conectados e população era de apenas $0,08$ dispositivo por pessoa. Com a popularização dos dispositivos ubíquos, esse número aumentou para $1,84$ dispositivo por pessoa em 2010 alcançando a marca de $12,5$ bilhões dispositivos conectados. Leftronic (2021) prevê que esse número chegue à 31 bilhões em 2021.

De acordo com Lee e Lee (2015), o sucesso da aceitação do IoT deve-se fundamentalmente ao uso de cinco tecnologias que são empregadas nos seus produtos e serviços: identificação de rádio frequência, redes de sensores sem fio (RSSF), *middleware*, computação em nuvem (*Cloud Computing*) e as aplicações IoT.

Em conjunto com as tecnologias utilizadas pelo IoT estão associados os dispositivos sensoriais, que são equipamentos que detectam e respondem a mudanças em determinados ambientes e podem atuar em diversos aspectos de sensoriamento, como: luminosidade, temperatura, movimento, umidade entre outros. As informações geradas por esses dispositivos são valiosas e estão, na maioria das vezes, conectadas a uma rede compartilhando esses dados com outros dispositivos e sistemas de armazenamento e análise. Os sensores estão em todos os lugares: industrias, casas, roupas, agropecuária, foguetes, entre outros e seu uso evoluiu sobremaneira principalmente com o surgimento da IoT.

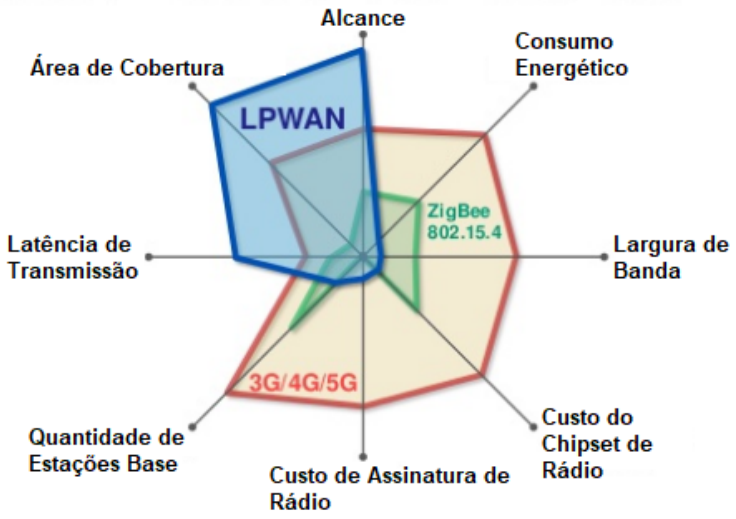
As diversas possibilidades que a IoT pode proporcionar em distintas áreas de pesquisa, necessariamente exigem meios de comunicação para o envio dos dados sensoriais. Essa necessidade é fundamental para a coleta dos dados, visto que não basta apenas que os sensores façam leituras e sejam monitorados localmente. É de suma importância que a informação esteja disponível para ser processada e analisada pelas aplicações, ficando a disposição para consulta dos usuários.

Visto o tamanho e a capacidade de processamento reduzido dos sensores, aliado ao fato de que esses objetos geralmente são instalados em locais de acesso dificultado, em especial na área rural, o uso mínimo de energia acaba sendo um requisito essencial para prolongar seu funcionamento. Vale lembrar que normalmente os dispositivos são alimentados por algum tipo de bateria, logo o potencial de suprimento energético é limitado. Desta forma, são desenvolvidas tecnologias de comunicação voltadas exclusivamente para

aplicações de IoT que tendem a otimizar o processo de comunicação e transmissão dos dados visando a eficiência energética.

Existem muitos meios de transmissão dos dados produzidos pelos sensores IoT para as aplicações e todos esses métodos tem suas vantagens e desvantagens. No entanto, o consenso sobre essa questão é que não existe uma solução universal aplicável a todos os cenários. Dependendo do modelo de infraestrutura utilizada e do volume de informações transmitidas, pode ser necessário uma rede com características de alta largura de banda, que impactaria no consumo energético. Outro cenário pode ser observado em que a infraestrutura faz uso de uma rede de celulares que possuem um longo alcance, porém com ônus financeiros devido aos planos exigidos pelas operadoras. A Figura 1 e a Tabela 1 trazem uma análise comparativa de uma série de fatores que podem impactar na escolha do modelo apropriado à comunicação IoT e, ainda mais essencial, o ponderamento desses atributos para uso na pecuária em que aspectos como área de cobertura e tecnologias de comunicação disponível são bem limitados.

Figura 1 – Atributos dominantes em redes IoT.



Fonte: Adaptado de Techplayon (2017).

Tabela 1 – Comparativo entre tecnologias de comunicação IoT.

Sistema de Comunicação	**Consumo de Energia**	**Custo de Operação**	**Largura de Banda**	**Alcance**
Celular	Alto	Alto	Alto	Longo
Wi-Fi	Alto	Médio	Alto	Baixo
Bluetooth	Baixo	Médio	Médio	Baixo
LPWAN	Muito Baixo	Baixo	Médio	Longo

Fonte: Adaptador de Rubio-Aparicio et al. (2019).

Na Figura 1, é possível observar os atributos em que as tecnologias LPWAN se destacam favoravelmente, tais como grande alcance, área de cobertura, baixo consumo energético e custo da tecnologia. Seu ponto fraco é a baixa largura de banda e a alta latência de transmissão dos dados. Na Tabela 1, vemos os atributos similares comparados novamente, com outras tecnologias.

Considerando o grande valor econômico que a agropecuária representa para o Brasil, é imprescindível o emprego de TIC que contribuam para o seu contínuo crescimento financeiro e tecnológico, garantindo o posicionamento como grande líder mundial no setor. Com esse propósito, a parceria entre a Universidade Federal de Mato Grosso do Sul[1] (UFMS) juntamente com Embrapa Gado de Corte[2], desenvolveram uma plataforma de IoT para auxílio na cadeia de produção de gado de corte, denominada "e-Cattle" (CARROMEU, 2019).

A plataforma e-Cattle é composta por um ecossistema digital completo com propósito de automatização do processo de produção de gado de corte e integração de dados sensoriais. É constituído por vários componentes como sensores para ambiente e animais, *middleware*, aplicação para nuvem e Interface de Programação de Aplicações (*Application Programming Interface* – API). Uma característica distinta da plataforma é o uso de variados protocolos de comunicação com os sensores, como: HTTP, LPWAN, M2M. Seu código fonte é aberto e pode ser acessado no GitHub[3], e uma abordagem mais detalhada a respeito da plataforma será feita no Capítulo 4.

Mediante o exposto, é possível afirmar que o uso de tecnologias que contribuam para uma maior eficiência na cadeia produtiva de gado de corte e que auxiliem na tomada de decisão dos produtores é fundamental para assegurar a liderança brasileira no setor. Outro fator importante a ser analisado é a escolha das tecnologias envolvidas, que devem ser apropriadas ao contexto do ambiente tornando-se ponto chave para o êxito da plataforma.

1.1 Justificativa

Um requisito fundamental para adoção de aplicações IoT no campo é o monitoramento de animais e do meio ambiente. Entretanto, o monitoramento de animais em fazendas com sistema de criação extensivo é um problema em lugares onde as áreas de pastagem chegam a centenas de milhares de hectares, sendo o Brasil um exemplo onde se encontra esse revés, devido a sua grande extensão territorial e economia fortemente relacionada à agropecuária. Contudo até mesmo os pequenos e médios produtores não estão livres desse transtorno.

[1] <https://www.ufms.br/>
[2] <https://www.embrapa.br/gado-de-corte>
[3] <https://github.com/e-cattle>

Após a realização de um estudo para levantamento de trabalhos que contemplassem o uso de aplicações IoT para gestão de produção de bovinos de corte, observou-se a dificuldade em descobrir conteúdo na área. Tendo em vista essa dificuldade, optou-se pela elaboração de uma solução que fizesse uso de tecnologias de baixo custo, fácil acesso e com ampla adoção da comunidade científica. Tais decisões são importantes para a adoção da solução, porque ela deve apresentar gastos com investimento mínimos. Porém, a agregação de valores, informação e retorno financeiro tendem a viabilizar sua utilização.

Propõe-se então o uso de redes de comunicação sem fio com sensores que tenham capacidade de transmissão dos dados por meio de redes LPWAN (*Low Power Wide Area Network*), para sistemas de monitoramento bovino. Essa escolha se deu pelo impacto que o problema do monitoramento promove na economia, sociedade e meio ambiente. Uma ampla pesquisa em torno dos protocolos de comunicação IoT que oferecem capacidade de transmissão de longo alcance, com características de mobilidade que são ideais ao contexto do cenário exposto. Em vista disso, o uso de tecnologias LPWAN apresentam ser uma solução ideal para a conjuntura apresentada, em conjunto com protocolos M2M (*Machine to Machine*) que foram desenvolvidos para comunicações entre dispositivos de modo a minimizar a interação humana e proporcionar menor sobrecarga de protocolo. Mais informações sobre essas tecnologias serão apresentadas nas Seções 2.1 e 2.2.

1.2 Objetivos

O objetivo geral desse trabalho é implementar um barramento de comunicação que utilize protocolos próprios para IoT, assim como meios de integração de diversos sensores utilizados na pecuária de precisão com a plataforma e-Cattle, proporcionando melhorias no alcance da comunicação entre os dispositivos.

1.3 Objetivos Específicos

- Arquitetar e desenvolver *drivers* de comunicação com sensores que utilizem protocolos IoT, como o LoRa e MQTT;
- Autenticar os sensores integrantes da plataforma;
- Persistir os dados sensoriais coletados no formato esperado pelo e-Cattle;
- Testar com aplicações reais.

1.4 Organização do Texto

Para melhor compreensão, este trabalho foi estruturado na forma de capítulos. O Capítulo 2 apresenta uma revisão teórica sobre o tema exposto. O Capítulo 3 expõe uma análise de trabalhos relacionados que utilizam IoT como solução para desafios de automação e monitoramento agropecuário de precisão. No Capítulo 4, é abordada a plataforma e-Cattle assim como as novas tecnologias que serão embarcadas. Por fim, no Capítulo 5 é descrito a conclusão desse trabalho de mestrado com relação as suas contribuições, dificuldades e trabalhos futuros.

2 Fundamentação Teórica

Este capítulo tem como foco apresentar os conceitos dos protocolos de redes utilizados em ambientes IoT para comunicação de sensores que vão compor a plataforma para monitoramento que está sendo desenvolvida. Também serão abordadas suas principais características, vantagens e desvantagens.

Na Seção 2.1, é realizada uma apresentação do que são redes LPWAN, com ênfase no protocolo LoRA. Na Seção 2.2 será discutido um modelo diferente de rede, onde é apresentado o conceito de redes M2M e foco no protocolo MQTT.

2.1 LPWAN

A utilização de tecnologias de acesso à Internet tradicionais como 3G, Ethernet e Wi-Fi podem ser aplicadas à IoT, porém o consumo de bateria dos dispositivos é o principal fator que determina a viabilidade ou não da utilização dessas tecnologias. Contudo é possível fazer uso de outras tecnologias de comunicação que apresentam consumo mais baixo.

Comunicações que não demandem de longo alcance podem fazer uso de redes ZigBee e *Bluetooth Low Energy* (BLE), por exemplo. Entretanto, a transmissão de dados em longas distâncias requerem uma outra categoria de rede que é conhecida como Rede de Área Ampla de Baixa Potência (*Low Power Wide Area Network* - LPWAN).

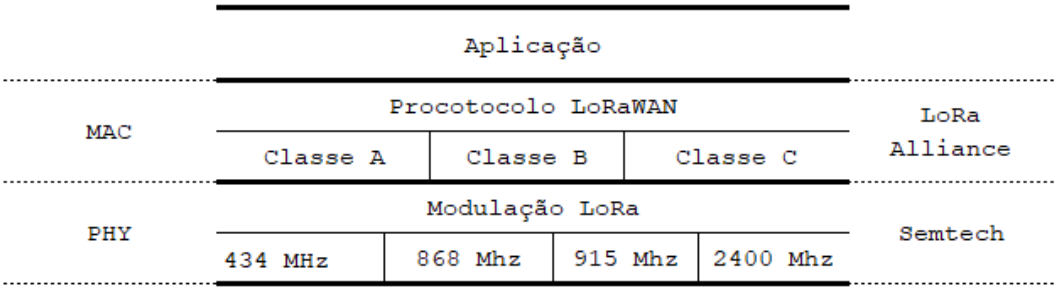
Redes LPWAN foram desenvolvidas exclusivamente para aplicações de IoT com propósito de conectar dispositivos geograficamente distantes, que tenham característica de baixo consumo de energia e com baixa taxa de transmissão, (RUBIO-APARICIO et al., 2019). Dentre as tecnologias LPWAN, as principais são: SigFox, LoRa e NB-IOT.

As semelhanças entre SigFox, LoRa e NB-IOT são o baixo consumo energético, topologia de rede em estrela e o fato de todas operarem em frequências abaixo de um Giga-hertz possibilitando a comunicação entre longas distâncias. Contudo, suas diferenças podem facilitar a escolha de uso de uma delas. Redes SigFox tem baixo custo de operação, visto que é necessário o uso de uma infraestrutura de rede existente de alguma empresa, embora sua taxa de transmissão seja bem limitada. O LoRa conta com algumas redes gratuitas, mas também é possível criar sua própria rede privada e sua taxa de transferência é razoável. Ao contrário das outras duas, o NB-IOT foi um meio que as operadoras de celular encontraram de ocupar frequências de banda não utilizadas no 4G, portanto existe um custo maior para uso da rede e dentre os três é o que possui maior largura de banda.

2.1.1 LoRa

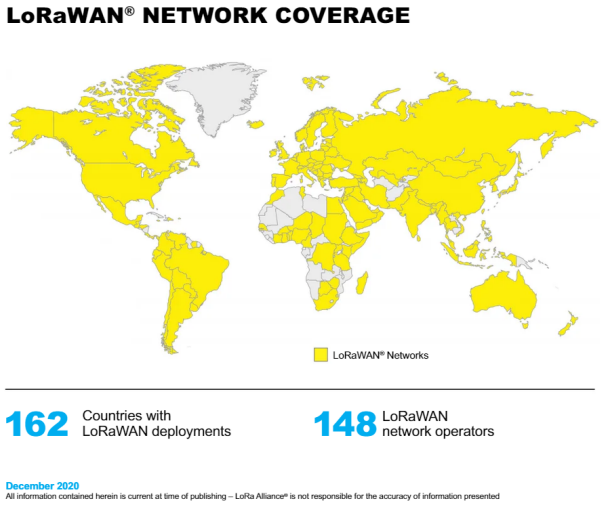
O LoRa (*Long Range*) foi criado por meio de uma *start up* francesa em 2010 e adquirida pela fabricante de semicondutores Semtech Corporation em 2012, que é a atual detentora da tecnologia. Sua arquitetura é dividida em duas camadas: LoRa, a camada física (PHY) e LoRaWAN (*Long Range Wide Area Network*) que define a camada de enlace (MAC), como observado na Figura 2. A LoRa Alliance é um padrão de código aberto que pode ser usada livremente em desenvolvimentos próprios para dispositivos e *gateways* (RUBIO-APARICIO et al., 2019).

Figura 2 – Arquitetura de camadas do protocolo LoRa.



Amplamente utilizado em aplicações que utilizam baixa taxa efetiva de transmissão, o LoRa utiliza a faixa sub-GHz, no Brasil 915MHz, o que permite alcançar distâncias de 5Km em áreas urbanas a 15Km em áreas rurais com baixo consumo de energia. Possui segurança com criptografia baseada no algoritmo AES-128b e apresenta como fatores de atração o seu custo aceitável e adesão de diversos fabricantes de *hardware* que a estão adotando em seus produtos. Como é possível verificar na Figura 3, a rede LoRa é amplamente utilizada ao redor do mundo.

Figura 3 – Implementação e usabilidade da rede LoRa no mundo.

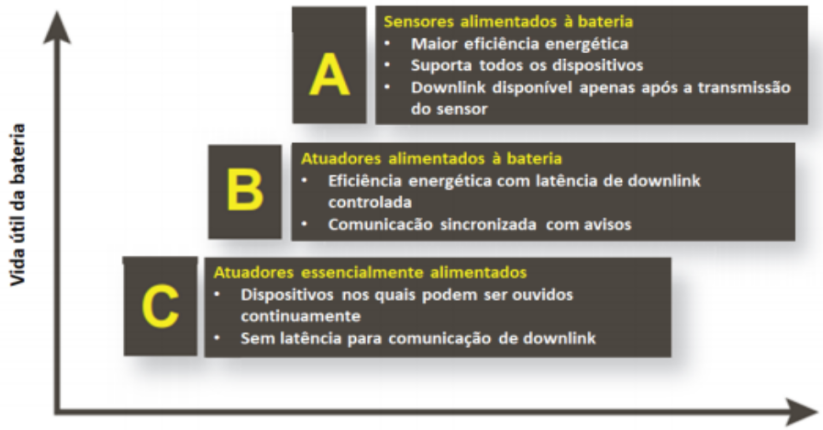


Fonte: Adaptado de LoRa Alliance (2020).

Redes LoRa possuem topologia estrela em que os *gateways* transmitem mensagens entre os dispositivos finais e um servidor de rede central. Os *gateways* são conectados ao servidor de rede por meio de conexões IP padrão e atuam como uma ponte transparente, simplesmente convertendo pacotes RF em pacotes IP e vice-versa. A comunicação sem fio aproveita as características de *Long Range* da camada física LoRa, permitindo um link de um único salto entre o dispositivo final e um ou vários *gateways*. Todos os modos são capazes de comunicação bidirecional, e há suporte para grupos de endereçamento *multicast* para fazer uso eficiente do espectro durante tarefas como atualizações de *Firmware Over-The-Air* (FOTA) ou outras mensagens de distribuição em massa.

Na página *web* da Lora Alliance existem três classes distintas de dispositivos finais para atender as necessidades diferentes entre os dispositivos, como pode-se observar na Figura 4.

Figura 4 – Comparação das classes de dispositivos por consumo de bateria.



Fonte: Adaptado de LoRa Alliance (2020).

Classe A - Dispositivos finais bidirecionais de menor potência: a classe padrão que deve ser suportada por todos os dispositivos finais LoRaWAN, a comunicação de classe A é sempre iniciada pelo dispositivo final e é totalmente assíncrona. Cada transmissão de *uplink* pode ser enviada a qualquer momento e é seguida por duas janelas de *downlink* curtas, dando a oportunidade para comunicação bidirecional ou comandos de controle de rede, se necessário (LORA ALLIANCE, 2020).

Classe B - Dispositivos finais bidirecionais com latência determinada: possui as mesmas características de comunicação dos dispositivos Classe A, porém com adição de sincronismo com a rede usando *beacons* periódicos que abrem "slots de ping" em horários programados. Isso fornece à rede a capacidade de enviar comunicações com uma latência determinada, mas com consumo adicional de energia no dispositivo final. A latência é programável em até 128 segundos para atender a diferentes aplicações e o

consumo de energia adicional é baixo o suficiente para ainda ser viável o uso de bateria (LORA ALLIANCE, 2020).

Classe C - Dispositivos finais bidirecionais de menor latência: reduz ainda mais a latência, mantendo o receptor do dispositivo final sempre aberto para que o dispositivo não esteja transmitindo (análogo ao modo de transmissão em *half duplex*). Com base nisso, o servidor de rede pode iniciar uma transmissão de *downlink* a qualquer momento, supondo que o receptor do dispositivo final esteja aberto, portanto sem latência. O compromisso é o consumo de energia do receptor (até 50 mW) e portanto a classe C é mais adequada para aplicações em que a energia contínua está disponível (LORA ALLIANCE, 2020).

Em vista disso, pode-se dizer que o LoRa tem muitos atributos que contribuem para sua adoção, seja ela motivada pelo fator financeiro, pela ampla área de cobertura e poder de infiltração em ambientes fechados ou até mesmo devido ao fato de grandes empresas de tecnologia apoiarem o seu desenvolvimento. Independente do motivo, redes LoRa vem apresentando boa aceitação do público e crescimento anualmente.

A integração de objetos físicos que capturaram dados sobre o ambiente ao seu redor e os compartilham com outros dispositivos, formam uma rede inteligente de "coisas" ou sistemas. Isso significa que as máquinas podem se comunicar e compartilhar informações sem a necessidade de interação humana, tornando esse processo em redes M2M (*Machine to Machine*). Na seção seguinte será apresentado a proposta de aplicabilidade da tecnologia M2M para o monitoramento de bovinos de corte.

2.2 Redes M2M

Com propósito de proporcionar um ambiente de monitoramento que possibilite agregar a maior variedade possível de dispositivos, foi observado a necessidade de inclusão na proposta o uso de protocolos M2M (*Machine to Machine*). Esse conceito de rede é aplicado geralmente a tecnologias de informação e comunicação (TIC) que são capazes de medir, entregar, processar e reagir as informações de maneira autônoma. As mensagens são trocadas diretamente entre os dispositivos e a atuação humana nessas redes deve ser a menor possível. Quando se fala em protocolos de comunicação para redes M2M, o MQTT (*Message Queuing Telemetry Transport*) é reconhecido como referência por sua simplicidade, eficácia e segurança (ANTON-HARO; DOHLER, 2014).

2.2.1 MQTT

O protocolo MQTT foi criado pela IBM no fim da década de 90, para transmissão de mensagens entre oleodutos e satélites. Yuan (2017) diz que o MQTT foi projetado para ser um protocolo escalável, e sua principal característica é desmembramento entre emissor e receptor da mensagem. Ou seja, a comunicação entre os envolvidos é assíncrona e não

tem problemas com latência ou atrasos. Por se tratar de um protocolo leve, ele possibilita a sua implementação em *hardware* que tem recursos de rede bastante restritos, como a largura de banda e alta latência.

O controle do roteamento baseado em IP pode ser um problema quando os nós da rede são móveis e com objetivo de minimizar esse tipo de adversidade, as arquiteturas baseadas em nome (*e.g.*: *Named Data Networking* (NDN), *Content Centric Networking* (CCN), e *Information Centric Networking* (ICN) tem sido adotadas. O MQTT pertence à categoria protocolos baseados na arquitetura ICN, e estudos demonstram sua capacidade de redução da sobrecarga de protocolo, assim como tem comprovado possuir uma comunicação altamente eficiente (YOKOTANI; SASAKI, 2016).

O protocolo HTTP tem sido amplamente utilizado na transferência de dados em RSSF. Contudo o uso desse protocolo em redes para Internet das Coisas pode causar sobrecarga, por conta da quantidade de troca de mensagens necessárias para comunicação entre os dispositivos. Yokotani e Sasaki (2016) fazem uma comparação entre o desempenho do HTTP com o MQTT. Nos experimentos realizados, o MQTT chega a ter RTT e latência até 80% menores em comparação ao HTTP. De acordo com Glaroudis, Iossifides e Chatzimisios (2020), o MQTT é o protocolo mais promissor e também o mais utilizado por pesquisadores quando o assunto é aplicações IoT para agricultura, eles analisam uma série de fatores que podem impactar na escolha do protocolo dependendo da contexto em que será utilizado, conforme Figura 5.

A comunicação do MQTT é realizada por meio de trocas de mensagens *Publish* e *Subscribe* (IBM, 2012). Quando algum dispositivo deseja enviar uma informação, ele efetua tal procedimento por meio de mensagens de *publish* enviadas para o *broker*. O *broker* por sua vez, é o elemento mediador de todas as comunicações. Por outro lado, quando um dispositivo deseja receber informação específica, ele faz o *subscribe* desses dados requisitando ao *broker*. O mecanismo de funcionamento do MQTT por ser observado na Figura 6.

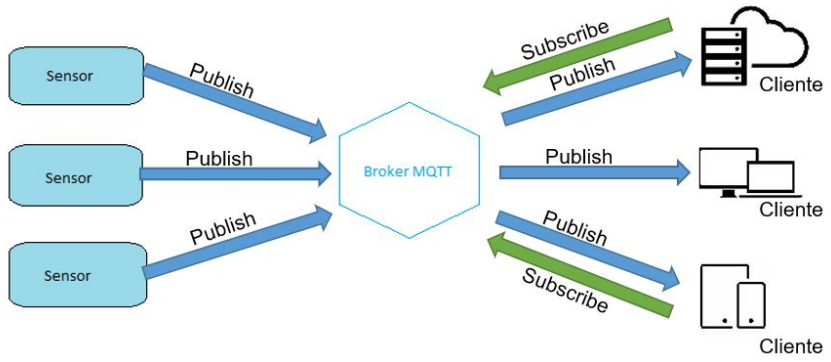
O MQTT possui três modos de transferência: QoS 0, QoS 1 e QoS 2, segundo (IBM, 2012). No modo QoS 0 a mensagem é enviada uma única vez e não existe confiabilidade de entrega, caso o cliente não receba essa mensagem ela será perdida. Esse comportamento é semelhante a transmissões UDP (*User Datagram Protocol*). O uso de QoS 1 implica que a mensagem deverá ser entregue pelo menos uma vez, exigindo confirmação de recebimento e tornando esse modo de transmissão confiável. Por fim, o QoS 2 exige confirmações nos dois sentidos, tanto de quem recebe as mensagens quanto de quem as envia. A escolha de qual QoS utilizar deve ser decidida com base no cenário onde a solução será aplicada avaliando o impacto na performance do sistema, sendo que a definição do QoS utilizado na comunicação é feita separadamente em duas partes: *Publishers* e *broker* bem como *broker* e *Subscribers*.

Figura 5 – Comparação de protocolos para aplicações IoT com base em indicadores de desempenho.

Key performance indicator	**Most promising protocol**				**Least promising protocol**
Latency					
Over a LAN	CoAP	MQTT QoS 0	AMQP	HTTP/REST	XMPP
Over a mobile network	MQTT QoS 0	CoAP	WebSocket	MQTT QoS 1	
Bandwidth consumption	CoAP	MQTT	AMQP and XMPP	DDS	HTTP/REST
Throughput	MQTT	DDS	CoAP	AMQP	XMPP
Reliability	MQTT	AMQP	CoAP	HTTP/REST	
Energy consumption	CoAP	MQTT	AMQP	HTTP/REST	
Developers' preference in recent IoT applications	MQTT	HTTP/REST	WebSocket	HTTP 2.0	CoAP, AMQP, XMPP, DDS
Researchers' preference in IoT agriculture applications	MQTT	HTTP/REST	CoAP		

Fonte: Adaptado de Glaroudis, Iossifides e Chatzimisios (2020).

Figura 6 – Funcionamento do MQTT.



2.3 Considerações Finais

O uso de tecnologia M2M adiciona maior heterogeneidade ao ambiente monitorado. Em conjunto com o LoRa, as possibilidades de agregação de novos dispositivos diferentes aumentam as chances de compatibilidade com a plataforma de monitoramento. Tendo em vista os estudos realizados das tecnologias que irão compor a plataforma de monitoramento bovino, abordaremos uma proposta no Capítulo 4.

Neste capítulo, abordamos os principais conceitos e características das tecnologias que serão utilizadas no desenvolvimento desse trabalho. O uso de tecnologias conceituadas

no mercado permite o desenvolvimento de um produto com maior solidez e, consequentemente, segurança aos usuários. O LoRa e o MQTT possuem esses atributos, tem uma base de desenvolvimento e pesquisa apoiada em grandes *players* da tecnologia da informação mundial.

3 Trabalhos Relacionados

Nesse capítulo uma análise de trabalhos relacionados será realizada com finalidade de fornecer um levantamento das principais ferramentas que utilizam tecnologias de IoT para solucionar problemas que envolvam monitoramento ambiental ou animal. Esses trabalhos possuem características comuns no quesito de monitoramento que fazem uso de sensores para detecção de eventos, mudanças no ambiente ou do animal. Portanto, o escopo para avaliação será analisar como e quais foram as tecnologias utilizadas na aplicação do sistema apresentado.

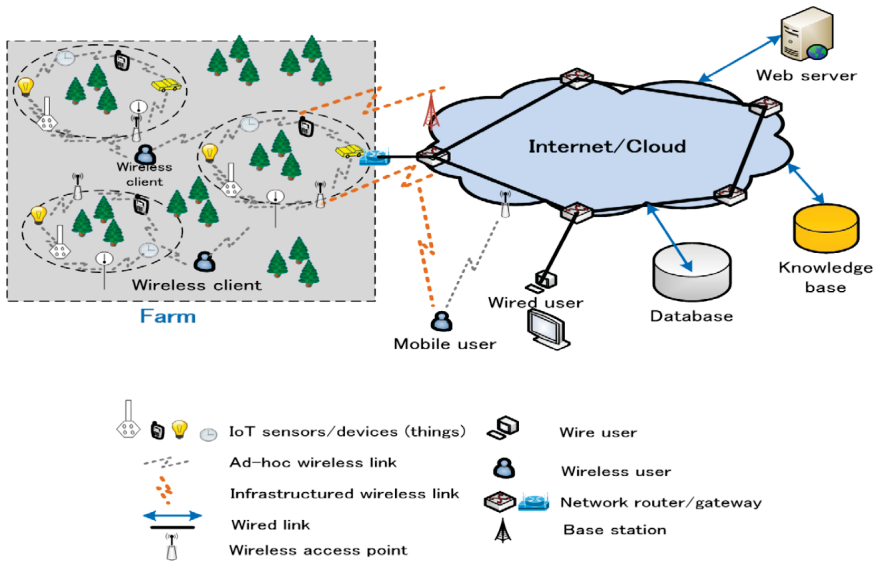
O propósito é apurar trabalhos que fazem uso de sensores para captação de informações sensoriais ambientais no domínio da agropecuária, avaliar como foram realizadas as soluções apresentadas e compará-los. Os aspectos comparativos são as escolhas das tecnologias para comunicação dos sensores e suas plataformas, *hardware*, arquiteturas, alcance e vazão. Os trabalhos aqui analisados foram selecionados com base na revisão sistemática descrita no Apêndice A.

3.1 Revisão Bibliográfica

Minh et al. (2017) propõem uma plataforma para ambiente agrícola, estruturada em camadas e que faz a integração de diferentes módulos como sensoriamento, comunicação e análise de dados que possibilite automação dos processos e conceda maior controle ao usuário. No cenário onde o experimento foi realizado eram cultivados milho e cogumelos. Foram elaborados kits de *hardware* para a plataforma que é composta por um *gateway* Raspberry Pi 3B, e sensores de umidade do solo e ar, pH, luminosidade, temperatura, entre outros. A comunicação entre os sensores e o *gateway* é realizada via enlaces de rede sem fio ad-hoc por meio do protocolo Zigbee e são alimentados por baterias. O *gateway* estabelece comunicação com uma aplicação *web* externa à fazenda por meio dos protocolos HTTP e MQTT. A arquitetura e a topologia de rede ser vista na Figura 7.

Mishra et al. (2019) evidenciaram a necessidade do aumento da produção de alimentos contrastando com a dificuldade do aumento das áreas de plantação e propõem um sistema de baixo custo associado a uma plataforma de monitoramento por meio de IoT que permitisse melhorar a produção. Foi adotada uma aplicação existente para IoT, chamada ThingSpeak que é composta tanto por serviços *web* quanto aplicativos para *smartphones* para o envio das informações coletadas. O conjunto de *hardware* é composto por NodeMCU com módulo Wi-Fi associado a sensores de temperatura, umidade do solo e ar, bombas de água. Todos os sensores são ligados ao NodeMCU que se comunica com as aplicações utilizando protocolos TCP/IP, HTTP ou MQTT por meio de redes Wi-Fi

Figura 7 – Visão geral da arquitetura do sistema proposto por Minh et al. (2017).



802.11b,g,n.

Ferdoush, Tahsin e Taher (2020) implementaram um sistema de *Smart Farm* com uso de redes WiMAX (*Worldwide Interoperability for Microwave Access*) para conectividade IoT, além de fazer uso da energia solar para atender aos requisitos de energia do sistema. São utilizados nove tipos de sensores diferentes para coletar dados em tempo real que ajudam os agricultores a decidir o momento apropriado para a irrigação e a colheita. O sistema de *hardware* é composto por um Arduíno Mega 2560 com os respectivos sensores acoplados a ele e possui topologia de rede em estrela. Após a aquisição dos dados as informações podem ser enviadas de duas maneiras, Zigbee ou WiMAX, que podem ser vistas em um sistema *web* ou via aplicativo móvel.

De acordo com Yoon et al. (2018) as tecnologias de IoT contribuem para superar as limitações dos sistemas de comunicação com fio e ajudam a ultrapassar as restrições de distância. Em seu trabalho, um sistema *Smart Farm* para uso agrícola em plantações cultivadas em estufas, foram empregados módulos de comunicação Bluetooth e LPWAN. Além disso, o sistema implementa as funções de monitoramento e controle usando MQTT, aumentando assim a possibilidade de desenvolvimento da IoT agrícola. Três diferentes abordagens de monitoramento foram utilizadas, uma com sistema de comunicação LPWAN via LoRa, outra com Bluetooh 4.0 (baixo consumo energético) e por fim, uma com sistema cabeado de comunicação RS485. Toda topologia de transmissão de informações é baseada na topologia de rede estrela. Cada abordagem tem sua própria rede física, mas cada rede converge para seu próprio *gateway*, que está conectado a um servidor com *broker* MQTT.

Jeaunita et al. (2018) propõe um design e a implementação de um sistema de estufa agrícola baseado em IoT para melhor rendimento das culturas. Sua proposta utiliza o protocolo MQTT com objetivo de minimizar a intervenção humana, detectando e

controlando automaticamente vários fatores climáticos. O sistema é composto por uma plataforma NodeMCU integrada com rede Wi-Fi. Toda automação é realizada no próprio NodeMCU que pode ter os seus parâmetros de monitoramento alterados e consultados via Internet por meio da aplicação Adafruit, que por sua vez, faz a comunicação com a plataforma via protocolo MQTT.

Kodali, Kuthada e Yogi Borra (2018) apresentam uma proposta de solução para os constantes problemas de escassez de recursos hídricos na agricultura. Um sistema de irrigação inteligente é utilizado para economizar tempo, dinheiro e energia do agricultor. O protótipo usa duas placas ESP32TTGO LoRa das quais uma é colocada no campo conectada a sensores como temperatura, umidade do solo e fluxo de água e a outra distante 5km é conectada à Internet usando o protocolo Wi-Fi. A comunicação entre as duas placas é feita por meio do protocolo LoRa. Os dados obtidos pela segunda placa são publicados na plataforma de *Cloud Computing* IBM Bluemix. Os dados são analisados pela plataforma Node-Red que envia uma mensagem motor de eventos do IBM Bluemix que pode acionar uma bomba de água automaticamente ou manualmente para irrigar os campos. Em trabalhos anteriores eram utilizados tecnologias como GPRS/GSM e Zigbee, porém elas adicionavam maior custo por conta do provedor desses serviços e são consideradas como protocolo desatualizado, respectivamente.

Jayageetha et al. (2020) desenvolveram uma solução para automação de fazendas. Algumas funções como análise da temperatura e umidade do solo podem disparar um gatilho para enviar um alerta ao agricultor, que pode ligar o sistema de irrigação remotamente por meio de aplicativo para *smartphone*. Outros exemplos são o monitoramento da incidência de luminosidade e alerta de animais em áreas não desejadas. Os sensores são ligados a um microcontrolador PIC (*Peripheral Interface Controllers*) responsável pela automatização dos procedimentos anteriormente descritos e que, segundo o autor, possui performance melhor em aplicações de tempo real. O sistema é atualizado pelo microcontrolador PIC que possui um módulo Wi-Fi para comunicação e sistema de alimentação por painel solar. Os dados são enviados ao usuário por meio de mensagens MQTT.

Para Islam et al. (2018) a agricultura de precisão requer integração de sensores, automação, recursos de rede e processamento de dados. Com o rápido crescimento da IoT no campo, os desafios de escalabilidade e gerenciamento de dados podem ser superados. Foi desenvolvido um sistema de monitoramento inteligente baseado em IoT, projetado um sistema de controle de irrigação com foco em eficiência energética e, por fim, um aplicativo cliente para *web* e Android. A escolha do LoRa foi determinante para superar as limitações de área de cobertura, permitindo comunicação de dados em tempo real. Tanto os sensores quanto os dispositivos de irrigação ficam espalhados pelo campo e se comunicam com o *gateway* LoRa com distâncias que podem chegar a 22 Km. Posteriormente, o *gateway* envia as informações para Internet por meio de conexão Wi-Fi a um servidor na nuvem que faz

o armazenamento em banco de dados. Os dispositivos que ficam espalhados pelo campo são compostos por Arduino Nano, enquanto o *gateway* é formado por um NodeMCU e ambos contém um transceptor LoRa com fonte de alimentação por bateria.

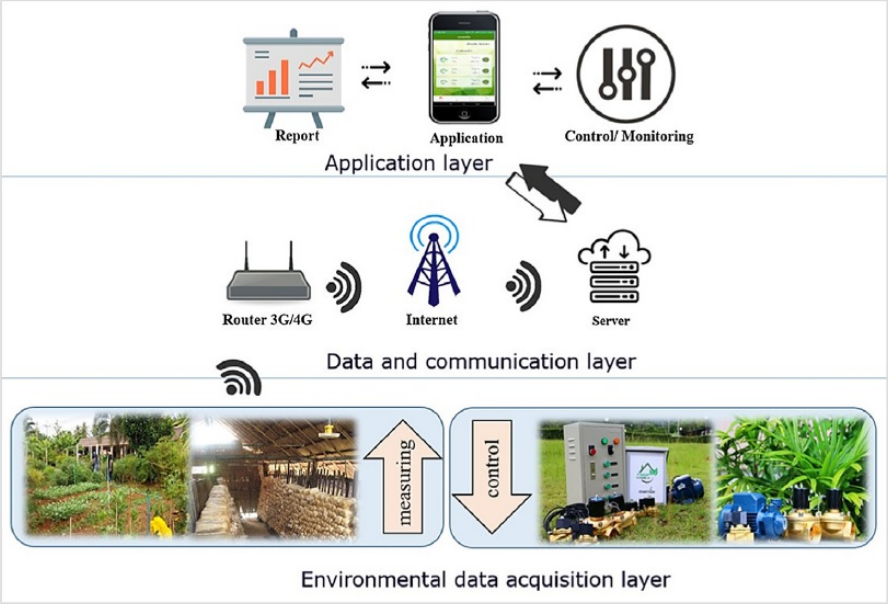
No país com maior rebanho do mundo de gado no mundo é fundamental que esses animais e o ambiente onde eles vivam sejam monitorados (Joshitha et al., 2021). O uso do LoRaWAN para monitoramento é importante pelas características que esse protocolo proporciona como transmissões de longo alcance e baixo consumo de energia. Adicionalmente são utilizados sensores de temperatura e umidade assim como um módulo de localização baseados em GPS (*Global Positioning System*) que ficarão contidos em uma coleira junto ao animal. Foram desenvolvidos dois dispositivos para o estudo proposto, um emissor e um receptor, ambos com propósito de baixo custo. O emissor é composto por um NodeMCU com chip LoRa SX1278, módulo de GPS, sensor de temperatura e fonte de energia. O receptor é constituído por um Raspberry Pi 3B e um módulo LoRa SX1278. A topologia de rede adotada no estudo é a estrela, em que os animais ficam com o dispositivo emissor enviando os dados para unidade central que é o receptor, que ao receber essas informações efetua o encaminhamento para uma plataforma IoT na Internet chamada Ubidots. Os dados sensoriais ficam salvos em um banco de dados da plataforma e disponíveis para visualização por meio de site da Internet.

De acordo com David et al. (2021) o cultivo de plantações em ambientes fechados vem crescendo ultimamente como uma alternativa ao modelo tradicional ao ar livre. Em seu trabalho é proposto um sistema que visa automatizar alguns processos desse novo paradigma. Para orquestração dos sensores e atuadores foi utilizado a ferramenta de desenvolvimento baseada em fluxo chamada Node-RED instalada na plataformas IBM Cloud e IBM IoT. Simulações virtuais foram realizadas para avaliar o sistema de automação em que eram monitorados e controlados fatores como temperatura, umidade, intensidade da luz e níveis de pH do solo, que são gerados pelo IBM IoT e enviados ao Node-RED por meio de mensagens MQTT. Ao final as informações ficavam disponíveis para visualização em uma aplicação móvel desenvolvida pelos autores.

Muangprathub et al. (2019) idealizou uma rede de sensores sem fio para gerenciar a irrigação de três tipos de culturas diferentes. O sistema proposto foi implementado em três partes: caixa de controle, aplicação web e aplicação móvel, e pode ser visto na Figura 8. A aplicação *web* é utilizada para gerenciamento dos dados e controle do sistema de irrigação e fica instalado no NodeMCU com conectividade Wi-Fi dentro da caixa de controle. Os sensores são conectados diretamente ao controlador que por sua vez transmite essas informações via Wi-Fi para aplicação *web*. Os resultados obtidos demonstraram melhoras no rendimento e qualidade dos alimentos cultivados, bem como redução dos custos de produção.

Germani et al. (2019) apresentam em sua pesquisa o design, implementação e

Figura 8 – Arquitetura do sistema de controle proposto por Muangprathub et al. (2019).



desempenho de uma arquitetura de *hardware* e *software* de IoT concebida para o monitoramento contínuo de bovinos em ambientes de pasto e celeiros. Foi adotado o LoRa para cobrir os diversos ambientes monitorados, assim como a configuração dos serviços *web* para executar armazenamento, análise e visualização de dados. Modificações no protocolo LoRaWAN foram implementadas para fornecer a habilidade de prevenção de colisão de pacotes com CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*).

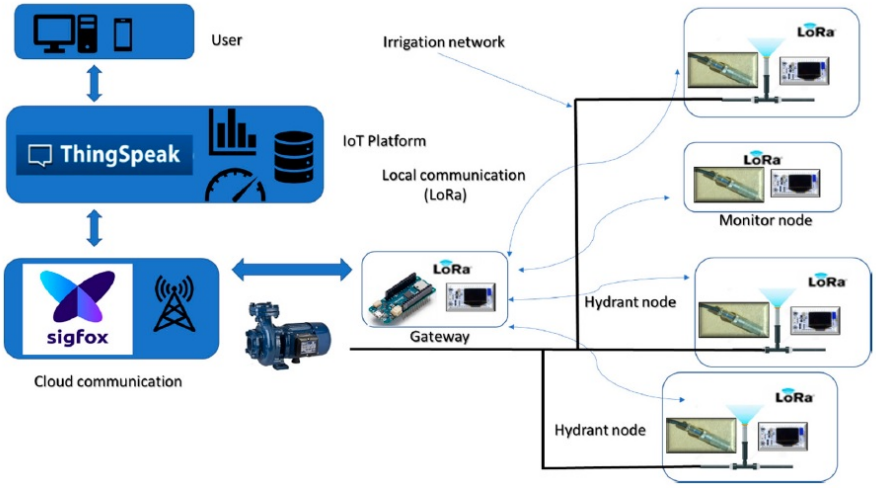
O *hardware* que compõe o *gateway* é formado por um Raspberry Pi 3B, modem 3G/4G e placa Adafruit Feather com rádio LoRa, uma outra versão é composta pelo mesmo modelo de Raspberry com adição de um Arduino Mega 2560 e sensores diversos. Uma terceira versão, diferente das outras duas descritas anteriormente que utilizam fonte de alimentação, é alimentada por bateria e placa solar.

As informações coletadas pelos *gateways* são transmitiras para um servidor de aplicação que possui uma API RESTful que grava os dados em um banco de dados PostgreSQL. Uma vez que os dados estão salvos no banco de dados, ele podem ser acessador por meio de um navegador web ou por clientes REST.

Fernandez-Ahumada et al. (2019) apresentam uma solução de baixo custo para irrigação automática com base no uso de comunicações RSSF e LPWAN, bem como na integração com plataforma IoT. O projeto fez uso de uma rede de nós baseada no microcontrolador ESP32-Lora para redes locais e conexão à Internet por meio da rede SIGFOX para acesso às plataformas IoT. A topologia de rede estrela foi adotada no trabalho e possibilita aos sensores utilizarem LoRa para enviar as informações ao *gateway* que por sua vez encaminha esses dados pela rede SIGFOX, conforme Figura 9.

Os *end devices* são constituídos de sensores relevantes as variáveis de irrigação e possuem uma placa de comunicação Heltec Wi-Fi LoRa 32. Os *gateways* por Arduinos MKR1200 que são interligados à uma placa Heltec com comunicação serial cabeada. A plataforma IoT ThingSpeak[1] foi escolhida por seu baixo requisito de infraestrutura. Os dados podem ser visualizados por meio de aplicação web e também podem ser analisados pela ferramenta matemática MATLAB®.

Figura 9 – Arquitetura do sistema proposto por Fernandez-Ahumada et al. (2019).



Nossa proposta visa ampliar os recursos de comunicação presentes na plataforma IoT de monitoramento pecuário de bovinos de corte de baixo custo, apresentado por (CARROMEU, 2019). Em seu trabalho, um *middleware* denominado BigBoxx, implementa diversas camadas que possuem papéis distintos e fundamentais de integração, possibilitando que o produto tenha desenvolvimento progressivo dentro da plataforma e-Cattle. O *hardware* é composto por um Raspberry Pi 3B vinculado com um *gateway* LoRa. Sua arquitetura dividida em barramentos de serviços possibilita a comunicação entre os diversos artefatos do ecossistema digital, tornando axiomático a função de cada camada. Algumas de suas propriedades são: topologia de rede estrela, barramento de comunicação com sensores via RESTful/HTTP e potencial para expansão para comunicação LPWAN. A plataforma atende as necessidades que a transformação digital vem provocando no agronegócio mundialmente que juntamente com sua ideologia de desenvolvimento contínuo, tornaram a escolha deste trabalho ideal para inclusão das tecnologias de comunicação IoT propostas.

Um comparativo dos trabalhos detalhados aqui pode ser conferido na Tabela 2 e também foram listados algumas características dos meios de comunicação usados na Tabela 3. Outras características também poderiam ser incluídas na tabela afim de enriquecer o comparativo(*e.g.* modelo do microcontrolador e do *chipset* LoRa), entretanto nem todos trabalhos detalhavam como foi realizado, ou não implementavam a mesma solução em comum.

[1] <https://thingspeak.com/>

Tabela 2 – Comparativo entre as tecnologias empregadas.

Trabalho	**Domínio do problema**	**Topologia**	**Protocolo IoT**	**Outros protocolos**
Minh et al. (2017)	Agricultura	Estrela	Zigbee	HTTPS, MQTT
Mishra et al. (2019)	Agricultura	Estrela	Wi-Fi	HTTP, MQTT
Ferdoush, Tahsin e Taher (2020)	Agricultura	Estrela	Zigbee/WiMAX	–
Yoon et al. (2018)	Agricultura	Estrela	LoRa	Bluetooth, MQTT, RS485
Jeaunita et al. (2018)	Agricultura	Estrela	Wi-Fi	MQTT
Kodali, Kuthada e Yogi Borra (2018)	Agricultura	Estrela	LoRa/Wi-Fi	–
Jayageetha et al. (2020)	Agricultura	Estrela	LoRa/Wi-Fi	–
Islam et al. (2018)	Agricultura	Estrela	LoRa/Wi-Fi	–
Joshitha et al. (2021)	Pecuária	Estrela	LoRa	GPS
David et al. (2021)	Agricultura	Estrela	–	MQTT
Muangprathub et al. (2019)	Agricultura	Estrela	Wi-Fi	–
Germani et al. (2019)	Pecuária	Estrela	LoRa/3G/4G	–
Fernandez-Ahumada et al. (2019)	Agricultura	Estrela	SIGFOX/LoRa	–

Tabela 3 – Características dos protocolos utilizados nos trabalhos analisados.

Protocolo	**Alcance**	**Taxa**	**Consumo energético**
3G/4G	35 Km	1 Gbps	Alto
LoRa	15 Km	50 Kbps	Muito Baixo
SIGFOX	50 Km	600 bps	Muito Baixo
Wi-Fi	50 m	600 Mbps	Alto
WiMAX	9 Km	70 Mbps	Alto
Zigbee	100 m	250 Kbps	Baixo

Nas tabelas citadas anteriormente, é possível observar a diversidade dos protocolos utilizados para o contexto da agropecuária, que podem variar bastante dependendo do cenário onde é aplicado. Dependendo da da situação, é necessária uma maior vazão de dados o que impacta no maior consumo energético. Em situações onde a necessidade é cobrir uma grande área de cultivo/criação, essa mesma abordagem não é viável pois os recursos de energia costumam ser escassos. Além disso, outros fatores podem interferir no desenvolvimento de soluções de monitoramento, como o custo de implementação/operação e a facilidade de uso.

3.2 Considerações Finais

Ao compararmos a solução proposta com os trabalhos correlatos descritos nessa seção, podemos encontrar similaridades entre eles como o uso de topologia estrela, pluralidade nos protocolos de rede e conectividade, assim como a utilização de artefatos de baixo custo que não afetem a qualidade e a capacidade do produto. Contudo, ao avaliarmos separadamente a plataforma e-Cattle com cada trabalho, temos um diferencial crucial que

é uma plataforma para monitoramento bovino (apenas 15% dos trabalhos abrangem o contexto da pecuária e tem relação com o contexto deste trabalho) além de possuir maior diversidade de integração com dispositivos IoT entre outros.

Todos os trabalhos optaram pelo uso de tecnologias que atendessem sua necessidade e ao cenário do problema explorado, podendo deduzir a inexistência de um modelo único que seja ideal a qualquer situação. Existem diversos métodos que podem ser aplicados ao problema e cada um vai adicionar pontos fortes e fracos à solução. Sendo assim, a finalidade desse trabalho é aliar o poder de comunicação de protocolos IoT ao trabalho proposto por (CARROMEU, 2019). Visto que, temos um problema porque os sensores e/ou plataformas podem estar distantes a quilômetros do seu receptor, um cenário em que o recurso de suprimento energético é escasso, há necessidade por protocolos que gerem um baixo *overhead* de comunicação, otimizando a performance da transmissão, e podemos induzir que os protocolos de comunicação IoT condizem com o paradigma exposto. Detalhes da implementação dos recursos propostos podem ser vistos no Capítulo 4.

4 Barramento multi-protocolo para comunicação de Sensores à Plataforma e-Cattle

A plataforma e-Cattle surgiu por uma necessidade da Embrapa Gado de Corte, para unificação dos dados sensoriais que já existem em outras aplicações do seu ecossistema digital de *Smart Livestock* (CARROMEU, 2019). O *middleware* BigBoxx é responsável por implementar o e-Cattle nas propriedades criadoras de bovinos de corte e realiza tarefas como: coleta, segmentação, validação, persistência e sincronismo dos dados e, quando possível, os exporta à nuvem. Dessa forma, as informações ficam unificadas e disponíveis para análise e tomadas de decisões em todo ciclo do processo do negócio (CARROMEU, 2019). Apesar de ser uma plataforma com ampla utilização de tecnologias IoT, o trabalho apresenta a limitação de comunicação restrita ao protocolo HTTP. O trabalho proposto é focado na inclusão de protocolos de comunicação LPWAN e M2M para agregar à plataforma diferentes tipos de comunicação, complementando assim a pluralidade de sensores admitidos.

Na Seção 4.1, é conceituado o funcionamento da plataforma e-Cattle e a sua organização em camadas. A Seção 4.2 descreve o que são os *Drivers* Dedicados e a sua função na arquitetura. A Seção 4.3 e 4.4 abordam respectivamente, o funcionamento do processo de comunicação das tecnologias MQTT e LoRa com o e-Cattle. Por fim, a Seção 4.5 apresenta as considerações finais acerca do capítulo.

4.1 Visão Geral da Arquitetura e-Cattle

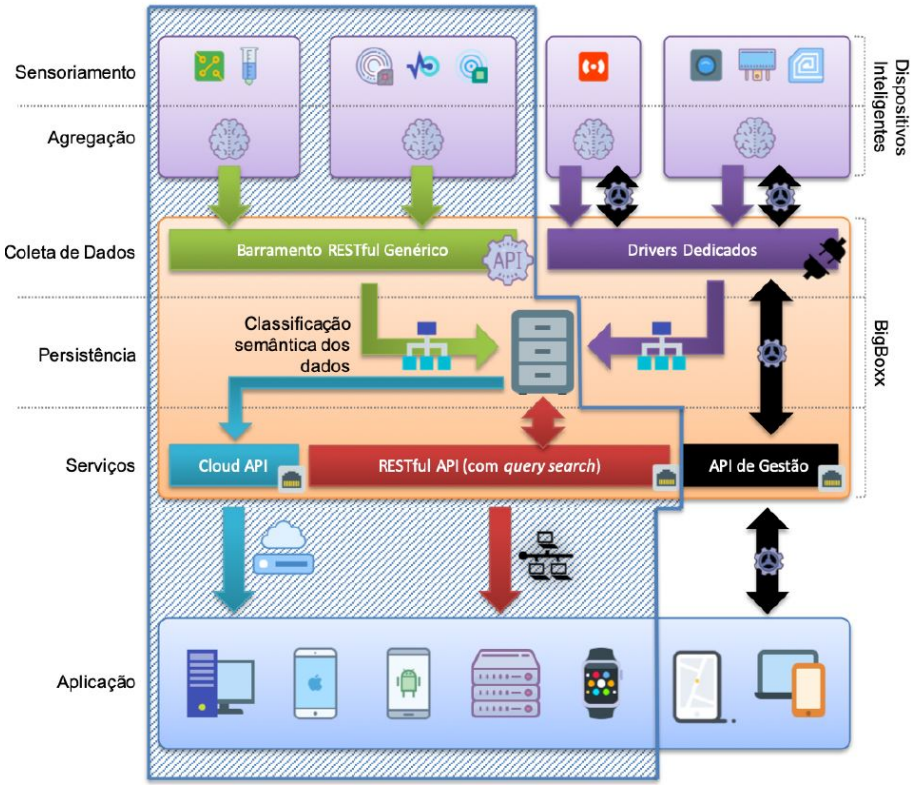
A plataforma e-Cattle é composta por diversas camadas com funções e propósitos bem definidos. Ao todo são seis camadas existentes que tem por finalidade manter a arquitetura simples, facilitando o desenvolvimento e futuras melhorias. Sua composição pode ser vista na Figura 10.

A camada mais externa, denominada Sensoriamento, integra os dispositivos de sensoriamento ambiental e animal, ou seja, componentes responsáveis por enviar as informações fisiológicas, de microclima, comportamentais ou contextuais que estejam monitorando.

A camada de Agregação desempenha o papel de coletor das informações vindas da camada de Sensoriamento e envia-os para a subsequente. É composta por coordenadores ou outros dispositivos auxiliares, como por exemplo um *gateway* LoRa ou um *broker* MQTT.

Na terceira camada, denominada Coleta de Dados, há duas interfaces para co-

Figura 10 – Arquitetura do e-Cattle, apresentando as 6 camadas e suas interações.



Fonte: Adaptado de Carromeu (2019).

municação com os dispositivos de sensoriamento ou coordenadores. A primeira, uma interface RESTFul para comunicações que fazem uso o protocolo HTTP (SANTOS, 2018). A segunda, uma interface para dispositivos que utilizam outros protocolos de comunicação para IoT, como MQTT, LoRa, ZigBee ou Sigfox. Nessa camada ocorre o controle dos dispositivos à plataforma, sua tarefa é avaliar se os dispositivos estão autorizados e homologados para estabelecimento da comunicação, estando aptos a enviar os dados para a plataforma. Além disso, os dados sensoriais são avaliados e tratados para com as normas definidas e, por fim, direcionados para a camada subsequente.

Na camada de Persistência, os dados sensoriais são gravados em um banco de dados NoSQL, devido à grande quantidade de informações produzidas pelos dispositivos sensoriais. Nessa camada são realizadas as funções pertinentes à padronização e sanitização dos dados sensoriais que ficam persistidos no banco Mongo[1], de acordo com um modelo de entidades definido por Silva (2018). Esses recursos são utilizados pelos *Drivers* Dedicados.

Na camada de Serviços, é realizada a comunicação com aplicações externas, quer seja para o envio dos dados para nuvem, para consulta dos dados no BigBoxx ou para acesso à interface para gerenciamento dos dispositivos inteligentes.

Por fim, temos a camada de Aplicação, onde diferentes aplicativos podem utilizar

[1] <https://www.mongodb.com/>

as informações recebidas para inteirar o produtor, auxiliando em suas tomadas de decisões, promovendo meios para automação de processos, produzindo novos indicadores e proporcionando uma melhor visualização da situação do negócio.

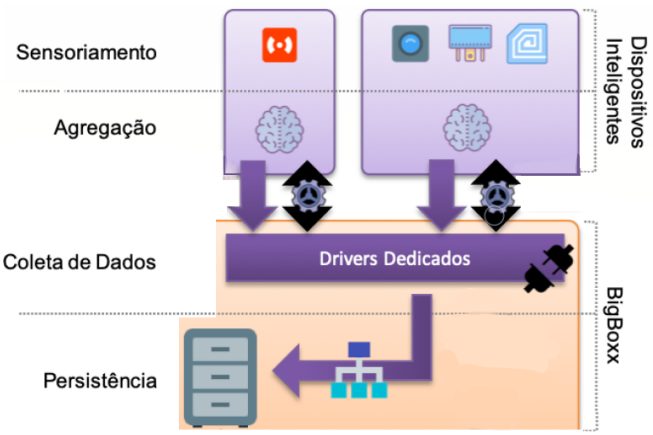
Os componentes da área hachurada em azul vistos na Figura 10 foram implementados por Santos (2018), Silva (2018), Cáceres (2020) e estão fortemente consolidados na plataforma. Essas funcionalidades existentes foram aproveitadas no desenvolvimento dos novos componentes da arquitetura e que serão melhor detalhadas posteriormente. Dentre as novas tecnologias reutilizadas estão a autenticação e validação dos dispositivos de sensoriamento e a sanitização e persistência das informações.

Atuantes na camada de Coleta de Dados, os *Drivers* Dedicados são programas elaborados com finalidade exclusiva de adicionar novas capacidades de comunicação com diversos protocolos de redes à plataforma e-Cattle. Diante do exposto objetivo desse trabalho, dois novos *drivers* devem compor a plataforma para comunicação com os protocolos LoRA e MQTT.

A área de atuação desse trabalho está focada em duas camadas primordiais: a camada de Agregação e a camada de Coleta de Dados. Essencialmente são nessas camadas que os protocolos de comunicação IoT interagem com a arquitetura, onde os dispositivos inteligentes integrantes da camada de Sensoriamento interagem com a camada de Agregação respectiva a sua tecnologia (*broker* MQTT ou *gateway* LoRa), e aguardam por sua homologação e autorização pelo *middleware* para então poderem transmitir os dados sensoriais. Os *Drivers* Dedicados que atuam na camada de Coleta de Dados tratam todas as informações recebidas pela camada de Agregação e disponibilizam esses dados para a camada de Persistência que faz armazenagem em banco de dados.

Um panorama simplificado da metodologia pode ser visto na Figura 11 e mais detalhes do processo de integração e comunicação serão apresentados nas seções seguintes.

Figura 11 – Camadas de atuação deste trabalho.

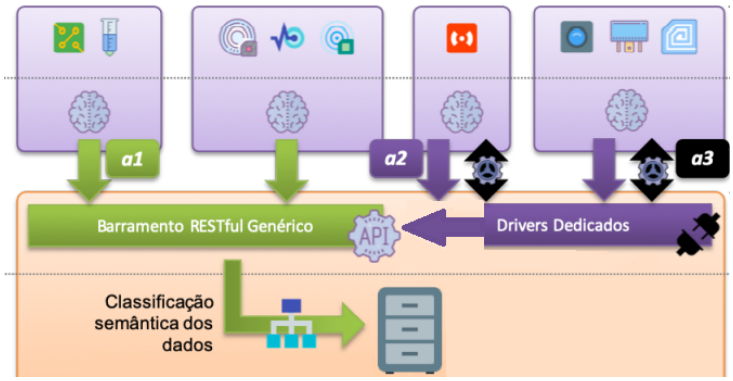


Fonte: Adaptado de Carromeu (2019).

4.2 *Drivers* Dedicados a protocolos de comunicação IoT

Inicialmente os *Drivers* Dedicados foram projetados para atuarem como pontes entre a camada de Dispositivos Inteligentes e a camada de Persistência (CARROMEU, 2019), como mostrado na Figura 11. Contudo, devido à decisão de manter o núcleo da plataforma mais simples ocasionou uma mudança na maneira como os *drivers* se comunicam, de modo que eles passaram a interagir diretamente com o barramento RESTful, conforme Figura 12. Consequentemente, as tarefas de tratamento e processamento dos dados coletados são realizadas de maneira centralizada e unificada. A escolha de desenvolvimento de *drivers* com esse comportamento deu-se pelo fato da facilidade na inclusão de melhorias futuras e novas funcionalidades não implicassem na necessidade de modificações nos *drivers* ou até mesmo desenvolvimento de um novo *driver*.

Figura 12 – Nova abordagem para os *Drivers* Dedicados da plataforma e-Cattle.



Fonte: Adaptado de Carromeu (2019).

O novo modelo de *drivers* adotado atua como um tradutor de linguagens, eximindo a responsabilidade de sanitização e persistência dos dados, visto que essa operação ficará a cargo das funcionalidades já existentes na plataforma e que passaram por diversos testes tornando-se componentes bem consolidadas.

Desenvolvido por Santos (2018), o barramento RESTful, responsável por receber os dados sensoriais, sanitizar e persistir em banco de dados, agora está diretamente ligado aos *Drivers* Dedicados, viabilizando o uso de suas funcionalidades por meio de API. Diante disso, toda a troca de mensagens entre o *driver* e a API ocorrerá mediante solicitações HTTP POST no *socket* TCP http://127.0.0.1:3000, ou seja, os *drivers* ficam incumbidos de traduzir os dados da camada de Sensoriamento para o formato *JSON* que é esperado pela API RESTful e vice-versa. Diferentes abordagens foram utilizadas para envio de mensagens do *middleware* aos dispositivos sensoriais, sendo que o *driver* MQTT retorna somente mensagens de erro, enquanto o *driver* LoRa não faz nenhuma notificação. Essas decisões foram tomadas com objetivo de manter a comunicação mais simples possível ao mesmo tempo que não exige que dispositivos alimentados por baterias permaneçam ligados

esperando mensagens de retorno.

4.2.1 Snaps

A plataforma e-Cattle foi idealizada como um sistema composto por artefatos de *software* modulares. Isso foi possível devido à distribuição Linux Ubuntu Core[2] que provê um ambiente integrado, confiável e seguro. O principal diferencial desse sistema operacional é o modo em que ele opera com um sistema de arquivos em modo de somente leitura e os programas instalados são entregues em forma de "Snaps" (similar a um contêiner) possibilitando atualizações independentes, confinadas, mais seguras e multi-plataforma. Os snaps são controlados, gerenciados e disponibilizados por meio da plataforma Web Snapcraft[3].

Os *snaps* que contêm os *drivers* para MQTT e LoRa da plataforma e-Cattle são ligados à plataforma de hospedagem de código e versionamento GitHub[4]. Assim, toda vez que o código sofre alguma alteração no repositório do GitHub, o Snapcraft automaticamente inicia a compilação e atualização dos snaps, vide a Figura 13. Caso ocorra algum problema na produção dos novos snaps, a última versão estável continua disponível na plataforma para *download*.

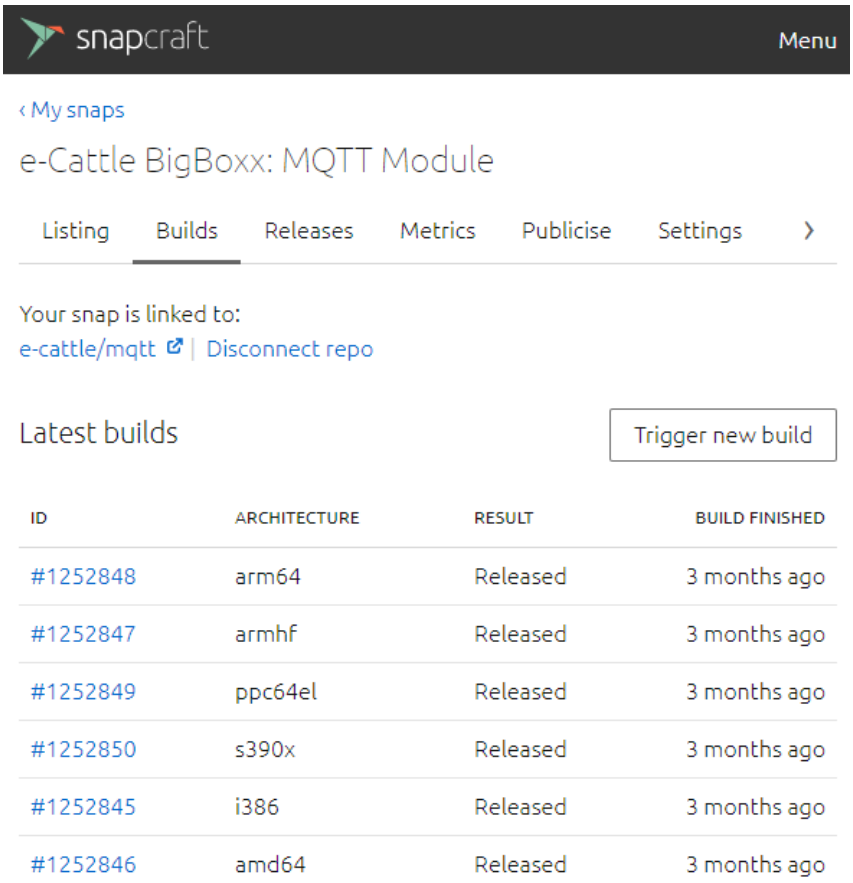
4.3 Processo de comunicação via MQTT

Em virtude da sua simplicidade, o MQTT é um protocolo amplamente utilizado para automações de ambientes M2M, sendo assim é fundamental que ele seja parte da arquitetura e-Cattle. Para que isso aconteça foi implementado um *driver* MQTT que atua na camada de Coleta de Dados proporcionando a integração de novos dispositivos à plataforma.

Para comunicação com protocolo MQTT é primordial o uso de um *broker*, conforme discutido na Seção 2.2.1, e arquiteturalmente ele está alocado na camada de Agregação dos dispositivos inteligentes. Sendo assim, foi escolhido o Mosquitto[5] como *broker* integrante do BigBoxx devido ao fato dele ser um *software* de código simples e aberto, amplamente utilizado pela comunidade e adequado para uso em todos os dispositivos, de pequenas placas com microcontroladores até servidores completos. Portanto, para que seja possível realizar uma comunicação com protocolo MQTT, são necessários duas componentes: o *broker* e o *driver* MQTT.

[2] <https://ubuntu.com/core/docs/what-is-ubuntu-core>
[3] <https://snapcraft.io/store>
[4] <https://github.com/e-cattle>
[5] <https://mosquitto.org/>

Figura 13 – Compilação dos *snaps* responsáveis pelo *driver* MQTT na plataforma Snapcraft.



O protocolo é baseado na publicação (*Publish*) e subscrição (*Subscribe*) de mensagens. As mensagens são transmitidas pelos dispositivos sensoriais por meio de tópicos não exigindo nenhuma configuração prévia no *broker*. Os tópicos requerem uma estrutura hierárquica na forma com que são definidos, análogo ao modelo de diretórios hierárquicos nos computadores. Todavia o MQTT oferece a capacidade para os clientes realizarem inscrições em tópicos específicos ou até mesmo grupos por meio dos coringas "+" ou "#" tornando possível ao *driver* MQTT seu acesso a todas as mensagens recebidas pelo *broker*.

Diante dos conceitos previamente expostos, os tópicos da plataforma e-Cattle foram padronizados em três tipos: (i) contrato, (ii) resposta e (iii) publicação. Cada tópico possui uma hierarquia e funções específicas no processo de comunicação do protocolo.

O primeiro modelo de tópico a ser informado é o tópico de contrato que é publicado pelo *endpoint*, por exemplo, um dispositivo sensor. Sua estrutura é definida como AAAA/Contract/CCCC, onde "AAAA" indica o local onde aquele dispositivo está fisicamente instalado; "Contract" informa que ele deseja aderir à plataforma e está enviando as informações básicas necessárias para essa finalidade; "CCCC" é composto pelo endereço de *mac-address* da placa de rede do equipamento. Os campos "AAAA" e "CCCC" são comuns nos três modelos de tópicos e exercem a mesma função. O conteúdo dessa mensagem deve

conter as informações necessárias para cadastro do dispositivo à plataforma, mais detalhes de como os contratos são compostos podem ser vistos no Apêndice B.

O segundo modelo de tópico é utilizado para o envio de respostas ao *endpoint* sendo necessário a inscrição no seguinte formato "AAAA/Response/CCCC". A diferença, nesse caso, é o diretório central "Response" que é empregado exclusivamente para envio de notificações do BigBoxx ao *endpoint*, ou seja, todos os dispositivos devem realizar *subscription* no seu respectivo tópico de resposta. Com intuito de minimizar ao máximo a quantidade de mensagens trocadas do *middleware* para os *endpoints*, optou-se por enviar mensagens informativas de erros (qualquer retorno que não seja *status code* 200 ou 201 do HTTP) e de aceite no envio de contratos.

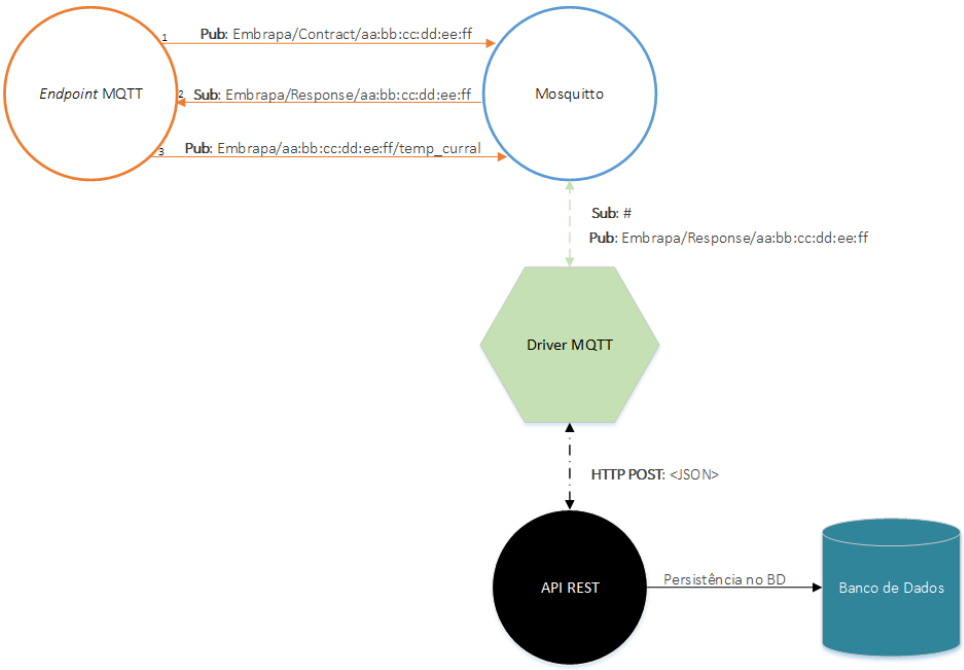
O terceiro modelo de tópico é o mais usado pois contém os dados das leituras sensoriais e tem o seguinte formato "AAAA/CCCC/YYYY", onde "AAAA" e "CCCC" seguem a mesma definição informada anteriormente e "YYYY" contém um "id" ou "tag" identificador do animal ou do dispositivo, esse último é opcional e pode ou não ser informado. Para o transmissão dos dados sensoriais é necessário apenas informar o valor lido como conteúdo da mensagem MQTT, todas outras informações necessárias são extraídas do tópico ou já foram informadas inicialmente na etapa de envio do contrato.

O fluxo na troca de mensagens ocorre excepcionalmente por meio dos três modelos de tópicos citados anteriormente, em que as mensagens são processadas pelo *driver* MQTT que fica incumbido de intermediar a conversão entre protocolo MQTT e encaminhá-la para API RESTful. O modo como as mensagens MQTT são trocadas na plataforma podem ser vistas na Figura 14.

Para melhor compreensão do funcionamento do *driver* MQTT, suponha que um novo *endpoint* deseja ingressar à plataforma. Primeiramente ele deve enviar uma mensagem de publicação no tópico de contratos com o formato e dados necessários para o mesmo. Visto que o *driver* tem acesso a todas as mensagens recebidas pelo *broker* ele processa a mensagem extraindo as informações e gera um pacote HTTP/JSON que é enviado à API RESTful. Enquanto o administrador do sistema não autorizar o ingresso do dispositivo, qualquer mensagem enviada por ele é descartada e o *driver* envia de volta ao dispositivo uma mensagem contendo o erro (*response code*) informado pela API.

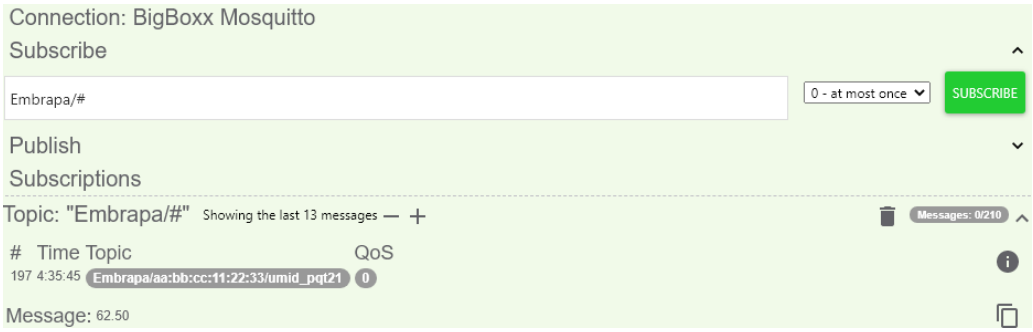
Autorizado o dispositivo, uma mensagem informativa é retornada da API ao *driver* que por sua vez encapsula a reposta no tópico de respostas com o respectivo *endpoint* e o envia. Nesse momento foi gerada uma senha no formato JSON *Web Token* (JWT) que é armazenada em banco de dados do tipo chave-valor no *middleware* e que deverá ser encaminhada em todas as publicações dos dados pelos sensores. Contudo, essa senha não é repassada ao dispositivo, ficando a cargo do *driver* realizar as consultas ao banco para então encapsular nas requisições à API. Dessa forma, é possível tornar o envio dos dados por parte dos sensores muito mais simples e compactos. A partir desse momento os dados

Figura 14 – Processo de comunicação de dispositivos que utilizam protocolo MQTT com o *driver* MQTT.



sensoriais podem ser enviados no tópico pertinente ao *broker* e todo processo de tradução ocorre de modo correlato a etapa de contrato. Um exemplo de publicação realizado por um sensor de umidade relativa do ar pode ser visto na Figura 15. Doravante as únicas mensagens que o *endpoint* poderá receber são as de indicação de algum erro, exceto das renovações de contrato.

Figura 15 – Publicação por MQTT de leitura de umidade relativa do ar realizada por um dispositivo com *mac-address* aa:bb:cc:11:22:33 da propriedade Embrapa .



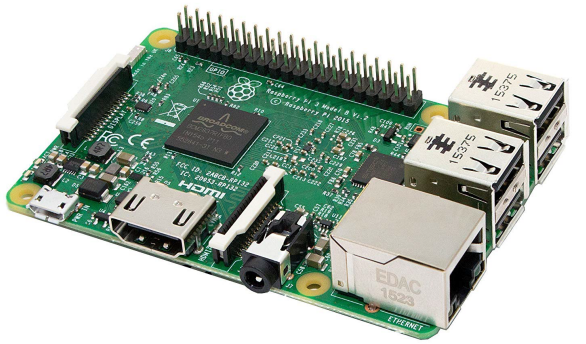
Em síntese, o *driver* desenvolvido atua como um componente tradutor que objetiva facilitar a integração de uma tecnologia bem consolidada como o barramento REST a novos aspirantes a integrantes, como o MQTT. Análogo à proposta de simplicidade do protocolo MQTT, o *driver* possui uma estrutura sucinta e concisa em sua atuação. No modelo desenvolvido é gerado um snap que possibilita sua utilização em sistemas operacionais e arquiteturas de diferentes tipos.

4.3.1 Testes com MQTT

Com objetivo de avaliar o impacto da implantação das soluções propostas com relação à limitação de hardware e também no que diz respeito à capacidade de suporte a comunicações simultâneas do *software* desenvolvido, testes foram realizados e informações sobre o uso de processamento, memória e rede foram coletados.

O *driver* MQTT assim como o sistema operacional Ubuntu Core 20.04.2 LTS foram instalados no micro-computador Raspberry Pi 3 Model B que dispõe do processador Broadcom BCM2387 equipado com ARMv8 Cortex-A53 com quatro núcleos operando a $1,2$GHz e GPU VideoCore 4 de dois núcleos à 300MHz, memória RAM de 1GB LPDDR2, possui conexões como Ethernet 10/100 BaseT, saída de vídeo HDMI 1.2 e 1.4, conectores GPIO 40 pinos, saída de áudio $3,5$mm, quatro conectores USB 2.0, interface serial para câmera MIPI 15 pinos, interface serial para monitor e *slot* para cartão de memória do tipo Micro SD. A Figura 16 exibe o modelo utilizado nos testes.

Figura 16 – Raspberry Pi 3 Model B.



Fonte: Adaptado de Carromeu (2019).

Para o monitoramento dos status do servidor, foi empregado o uso da ferramenta auxiliar dedicada a essa função, chamada RPi Monitor[6]. Ademais o sistema operacional utilizado foi Ubuntu 20.04.2 LTS (ARM64) com todas as atualizações aplicadas e tendo em vista uma melhor observação do impacto das soluções, apenas o SNAP do BigBoxx Kernel e do MQTT foram instalados, mantendo o cenário de teste mais simples e sucinto possível reduzindo as chances de outras aplicações impactarem nos resultados.

Para testar o *driver* MQTT foi necessário desenvolver uma nova aplicação que recebesse um conjunto de dados como entrada e enviasse essas informações ao *broker*. No primeiro caso de teste, um conjunto com dados reais coletados por balanças de peso do animal foi utilizado para simular um caso hipotético no qual doze balanças informavam suas leituras a cada minuto, distribuídas uniformementes em intervalos de cinco segundos (totalizando 720 pesagens por hora), por um período de simulação de três

[6] https://github.com/XavierBerger/RPi-Monitor

dias consecutivos. Dessa forma foi possível obter os seguintes resultados de consumo de processador (Figura 17), memória (Figura 18) e rede (Figura 19), nas quais estão registrados a carga de uso dos quatro núcleos da CPU (onde o valor "0" que dizer nenhuma carga de processamento e o valor "4" quer dizer que os quatro núcleos estão sendo completamente utilizados), quantidade de memória livre e disponível (que foi otimizada em forma de cache) em megabytes (MB) e volume de dados trafegados na interface de rede em kbps, todos esses parâmetros em relação ao tempo de três dias do experimento.

Figura 17 – Primeiro experimento, informações do uso de processamento do *driver* MQTT na simulação com doze leituras por minuto.

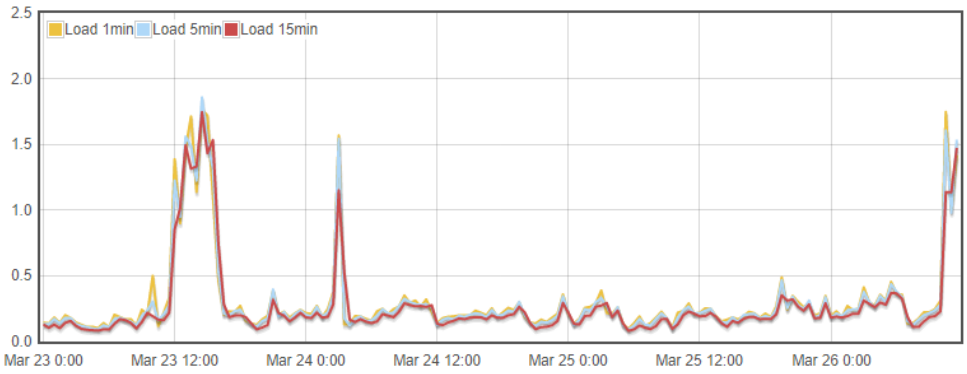
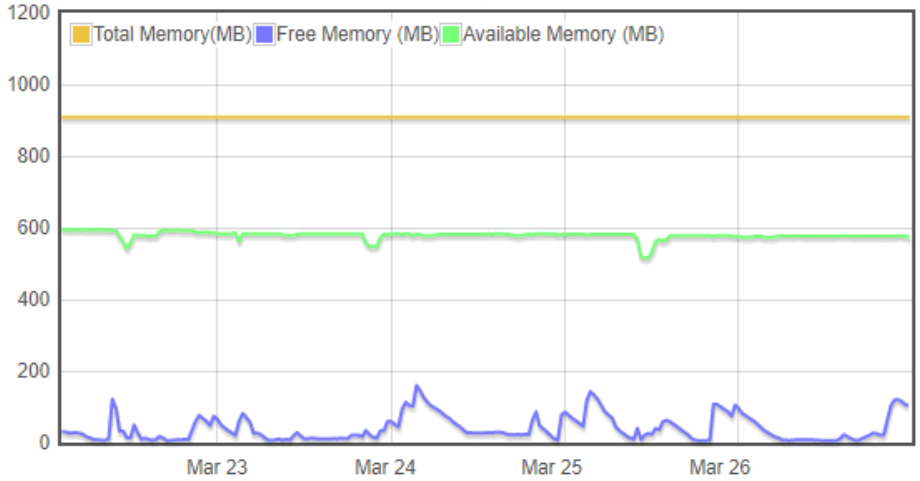


Figura 18 – Primeiro experimento, informações do uso de memória do *driver* MQTT na simulação com doze leituras por minuto.



O segundo caso de teste foi realizado variando os parâmetros de quantidade de sensores e dados sensoriais, de modo que, foram simulados os envios de dados de dez dispositivos informando temperatura do animal, temperatura ambiente e umidade relativa do ar a cada dez segundos, com um total de 10.800 dados sensoriais por hora. Igualmente ao primeiro teste, nesse caso também foram empregadas informações obtidas de um conjunto dados real. O resultado dessa experimentação pode ser observado nas Figura 20, Figura 21 e Figura 22.

Figura 19 – Primeiro experimento, informações do uso de rede do *driver* MQTT na simulação com doze leituras por minuto.

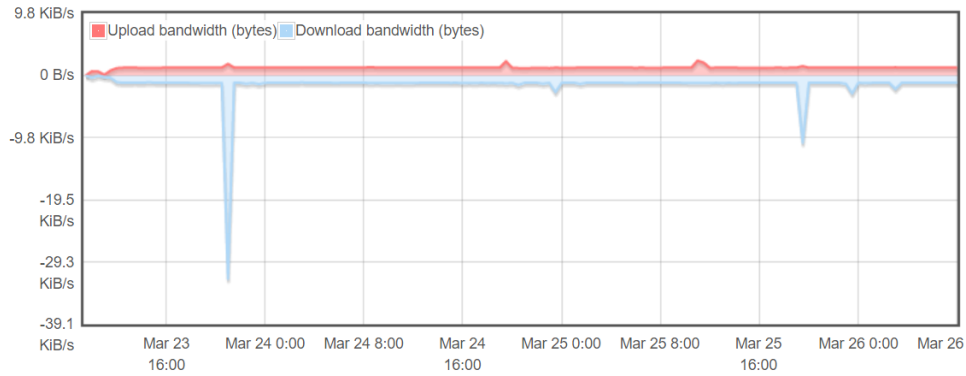


Figura 20 – Segundo experimento, informações do uso de processamento do *driver* MQTT na simulação com 180 leituras por minuto.

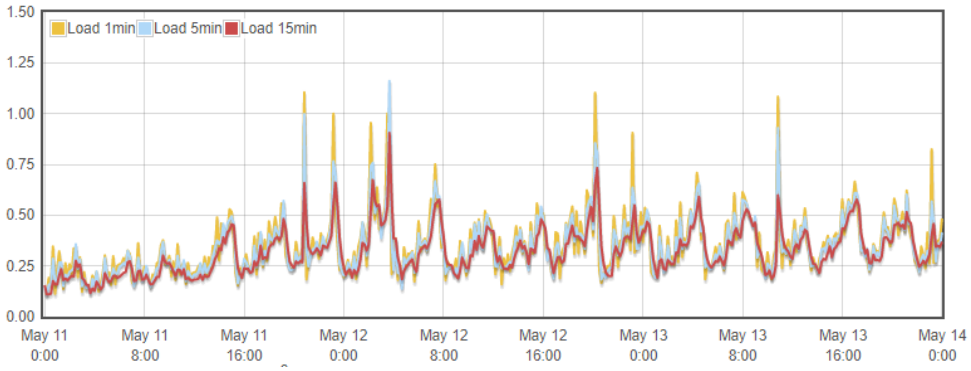
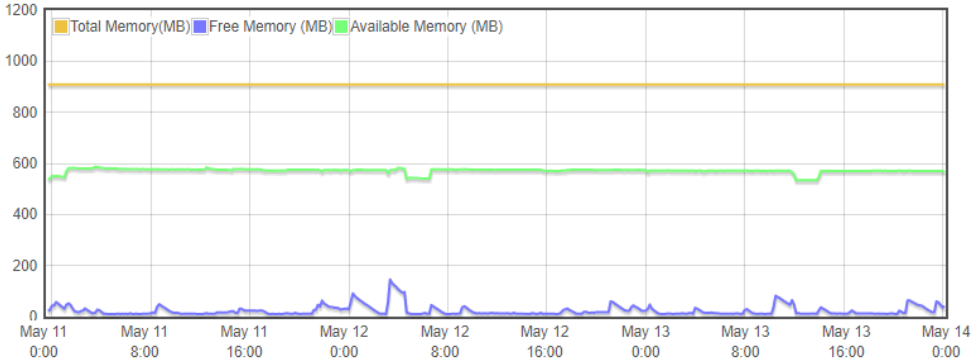


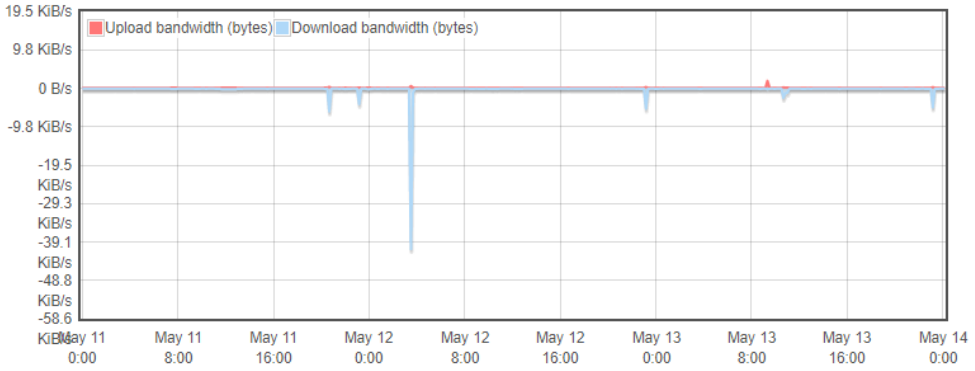
Figura 21 – Segundo experimento, informações do uso de memória do *driver* MQTT na simulação com 180 leituras por minuto.



Em ambos os casos de testes a escolha do número de vezes em que os sensores enviam as leituras ao BigBoxx foi definida com base em valores que pudessem ser uma realidade tanto para o pequeno quanto o grande produtor de bovinos de corte. Desta maneira é possível observar separadamente os impactos em ambas situações.

Os resultados das experimentações realizadas demonstraram que o *driver* MQTT obteve um bom desempenho, conseguindo tratar as mensagens sem ocasionar sobrecargas ao sistema. De modo geral tanto no primeiro caso de teste quanto no segundo, o uso

Figura 22 – Segundo experimento, informações do uso de rede do *driver* MQTT na simulação com 180 leituras por minuto.



de CPU ficou abaixo de 50% de um núcleo, a quantidade de memória disponível ficou em torno de 600MB e o uso de rede abaixo de 5kbps. Os picos capturados nos gráficos de uso de CPU e rede provavelmente foram provocadas por algum processo do sistema operacional que estava consumindo esses recursos visto que a maior parte do tempo não foi observado esse comportamento.

4.4 Processo de comunicação via LoRa

O *diver* Lora está presente na camada de Coleta de Dados da arquitetura do e-Cattle, e a sua função é prover uma nova funcionalidade de comunicação com dispositivos integrantes da categoria LPWAN especificamente o LoRa.

Por se tratar de um protocolo cujas características de funcionamento correspondem às necessidades do ambiente onde a plataforma e-Cattle atua, o *driver* LoRa atuará como um facilitador na integração entre essas tecnologias. A utilização da aplicação de código aberto denominada ChirpStack[7] foi necessária para provimento de uma solução unificada que possibilita a gerência e integração de redes LoRaWAN e é divido em três módulos: ChirpStack Gateway Bridge, Chirpstack Network Server e ChirpStack Application Server. O funcionamento de cada componente será discutido a seguir.

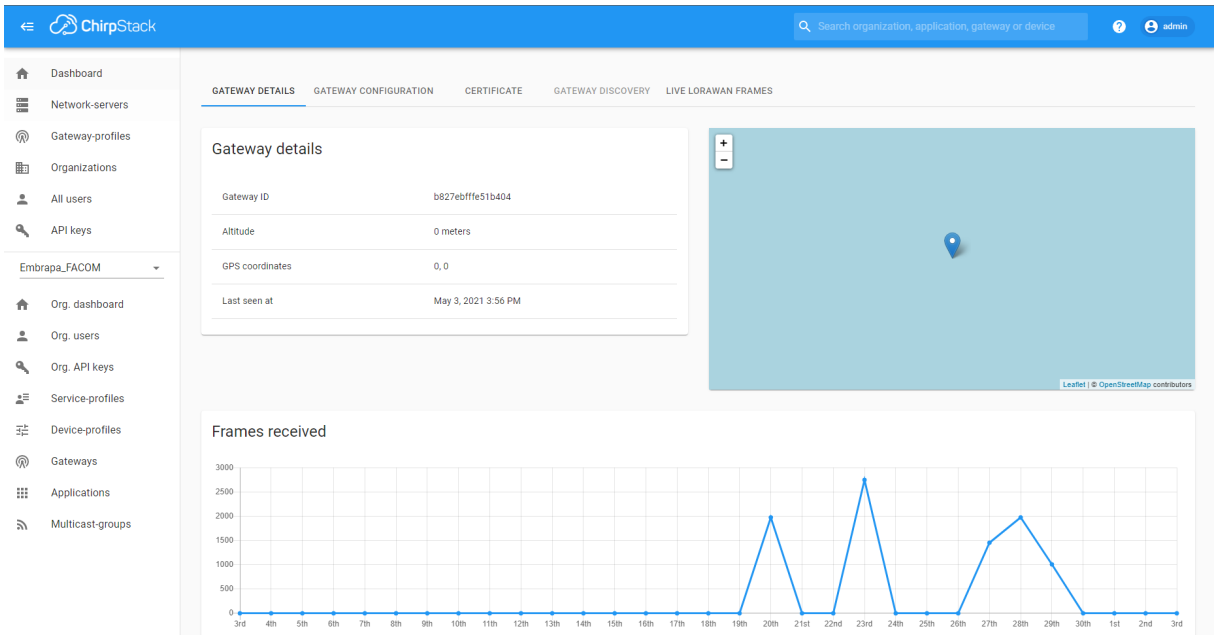
O módulo ChirpStack Gateway Bridge atua na comunicação direta com o(s) gateway(s) LoRaWAN, recebendo os encaminhamentos dos pacotes UDP, formatando-os em mensagens JSON e enviando-os para o componente seguinte. Conseguinte o componente ChirpStack Network Server recebe esses dados e atua como um servidor de rede LoRaWAN, lidando com a de-duplicação de mensagens, cuidando da camada MAC LoRaWAN e realizando agendamentos para transmissão de mensagens de *downlink*. Por fim, o ChirpStack Application Server fornece uma interface Web para gerência de toda essa plataforma, além de cuidar das requisições de *join* dos dispositivos integrantes da rede, criptografia dos

[7] <https://www.chirpstack.io>

payloads e propicia acesso via API RESTful, gRPC ou MQTT para serviços externos.

Na componente ChirpStack Application Server, são realizadas todas as configurações pertinentes à integração dos dispositivos LoRa, incluindo a criação das aplicações de rede que representam os modelos de dados sensoriais aceitos pelo banco de dados do BigBoxx. Além disso, as chaves criptográficas utilizadas para comunicação são definidas aqui, possibilitando que dispositivos LoRa se comuniquem com o BigBoxx por meio de ativação OTAA. Na Figura 23 é possível observar informações pertinentes ao *gateway* Lora gerenciado pela aplicação ChirpStack, que apesar de estar disponível para acesso ao usuário, não é necessário nenhuma interferência do mesmo.

Figura 23 – Dashboard do ChirpStack para visualização das mensagens recebidas pelo *gateway* LoRa.

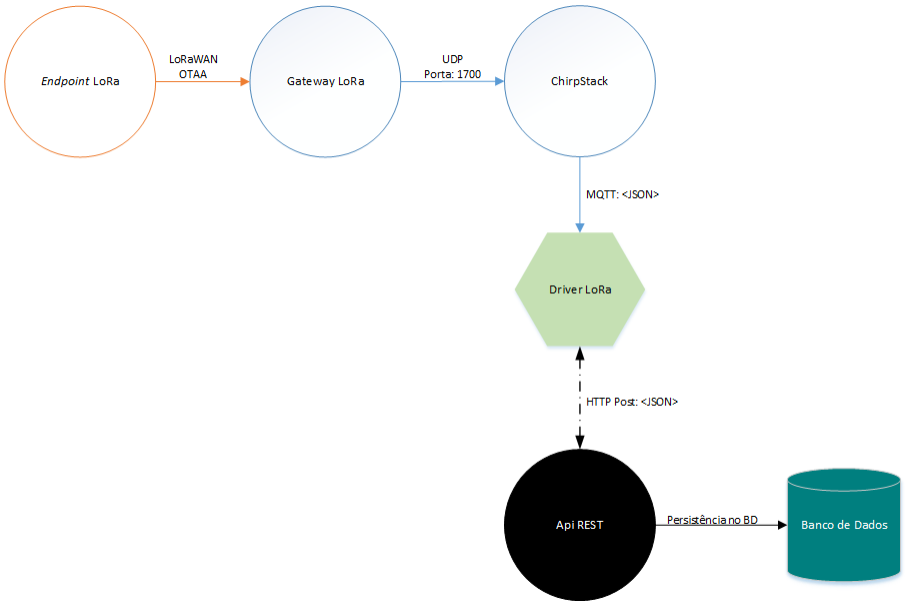


O *driver* LoRa comunica-se diretamente com a o ChirpStack Application Server via MQTT e com o barramento genérico RESTful. Sua função é lidar com as mensagens recebidas pelo ChirpStack, encapsulando-as no formato esperado pela API da camada de Coleta de Dados. Dessa forma, todo o trabalho de tratamento das informações é transferido para um ponto unificado e consolidado. A Figura 24 exemplifica o comportamento exposto.

Além das funções citadas anteriormente, o *driver* também está encarregado de cuidar do envio de contratos (para mais detalhes sobre os contratos veja o Apêndice B) dos *endpoints* e certificação de que existem criados no ChirpStack todos os modelos de dados aceitos pelo e-Cattle. A adoção de tais medidas foi necessária para minimizar as dificuldades de configuração dos *endpoints* tornando o uso do recurso mais acessível.

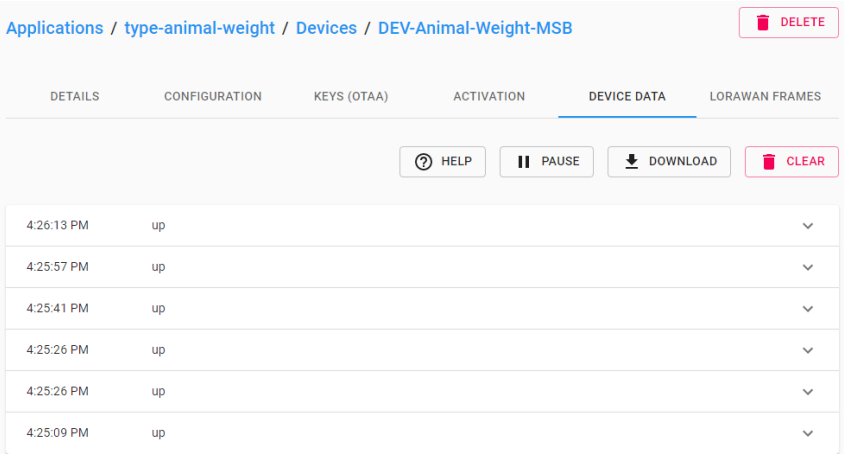
De início, a primeira função executada pelo *driver* LoRa é a checagem das aplicações criadas no ChirpStack, onde é comparado um conjunto de aplicações oficiais suportadas

Figura 24 – Processo de comunicação por meio do protocolo LoRa.



previamente definidas no próprio *driver* juntamente com a relação de aplicações existentes cadastradas no ChirpStack. Caso alguma aplicação não exista, o *driver* procede com a criação do mesmo. São as aplicações que definem o tipo de dado sensorial que o *endpoint* está enviando, portanto cada tipo de dado sensorial possui uma aplicação vinculada, na Figura 25 é possível visualizar o recebimento de mensagens da aplicação para medições de temperatura do ar. Após essa etapa, o *driver* dá início de fato ao processo de recebimento dos dados sensoriais por intermédio do ChirpStack que fornece essas informações via MQTT com conteúdo em formato JSON.

Figura 25 – Tela de acompanhamento em tempo real das mensagens recebidas pelo gateway LoRA para aplicação de temperatura do ar.



Visto que os *endpoints* realizam somente o envio das leituras dos sensores, o *driver* LoRa fica incumbido de cuidar da geração dos contratos desses dispositivos. Para toda mensagem recebida, primeiramente é checado se existe um token de contrato válido. Se a

verificação ocorrer com sucesso, são extraídos os dados do JSON e encapsulados em uma nova mensagem também JSON, porém no formato esperado pela API RESTful. Se esse dispositivo não possuir um token válido, o *driver* gera um contrato e encaminha para a API, repetindo esse procedimento sempre que esse error ocorrer. As próximas mensagens possuirão um token válido que possibilitará tratar os dados do dispositivo de sensoriamento pelo *driver* e consequentemente recebimento pela API.

4.4.1 Testes com LoRa

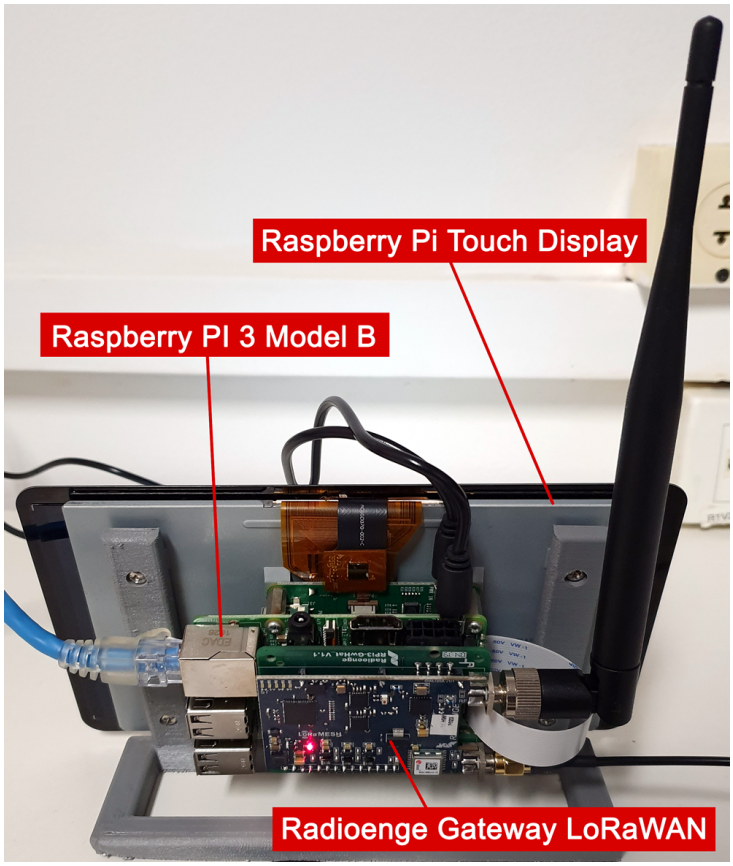
Análogo a etapa de condução dos testes com *driver* MQTT (subseção 4.3.1), uma nova sequência de testes foi realizada no *driver* Lora, com intuito de garantir a robustez dos programas desenvolvidos, assim como avaliar os impactos na performance do hardware. Foi utilizada a ferramenta de monitoramento RPI Monitor para coleta dos dados estatísticos e produção dos gráficos dessas informações. Além disso, o sistema operacional utilizado foi o Raspiberry Pi OS[8] (ARM64) com todas atualizações aplicadas. Diferente do primeiro teste, nesse caso a utilização de outro sistema operacional se deu pelo fato de uma incompatibilidade entre o *gateway* LoRa e o Ubuntu Core que não foi possível solucionar em tempo hábil, mesmo assim foi empregado o uso dos SNAPs do BigBoxx Kernel e também do *driver* Lora.

Outra diferença em relação aos testes do MQTT, está na utilização do *gateway* LoRa da empresa Radioenge que implementa o protocolo de comunicação LoRaWAN e padrão de criptografia AES. O *gateway* detém um *data rate* adaptativo entre 980bps e 21900bps, com topologia de rede estrela. O padrão de rádio frequência do LoRA que foi adotado no Brasil utiliza a banda de 915MHz e portanto o *transceiver* utilizado é o Semtech SX1257 e a sua potência de transmissão é de +27dBm com sensibilidade de recepção de até −137dBm. Na Figura 26, pode-se observar a integração do *gateway* ao Raspberry Pi.

Para testar o *driver* LoRa foi necessário desenvolver uma nova aplicação que recebesse um conjunto de dados como entrada e enviasse essas informações ao *driver*. No primeiro caso de teste, um conjunto com dados reais coletados por balanças de peso do animal foi utilizado para simular um caso hipotético no qual doze balanças informavam suas leituras a cada minuto, distribuídas uniformemente em intervalos de cinco segundos (totalizando 720 pesagens por hora), por um período de simulação de três dias consecutivos. Dessa forma foi possível obter os seguintes resultados demonstrados nas Figura 27, Figura 28 e Figura 29, onde estão registrados a carga de uso dos quatro núcleos da CPU em porcentagem, a quantidade de memória disponível em porcentagem e o volume de dados trafegados na interface de rede em kbps, respectivamente, todos esses parâmetros em relação ao tempo de experimentação de três dias.

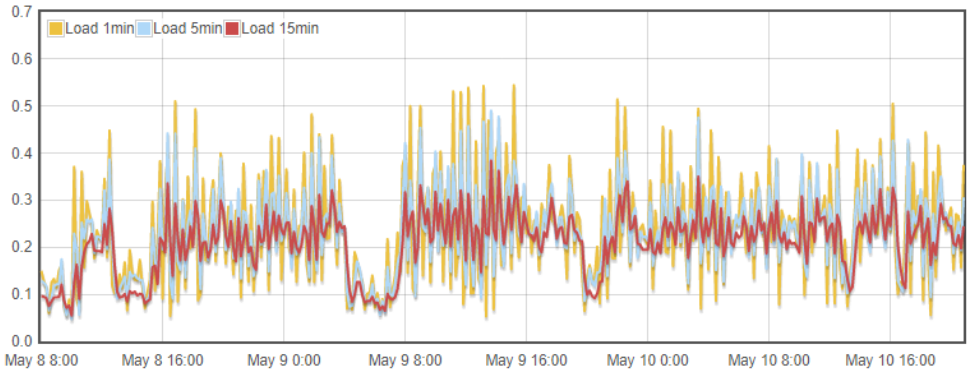
[8] <https://www.raspberrypi.org/software/operating-systems/>

Figura 26 – *Middleware* BigBoxx com *Gateway* LoRaWAN.



Fonte: Adaptado de Carromeu (2019).

Figura 27 – Primeiro experimento, informações do uso de processamento do *driver* LoRa na simulação com doze leituras por minuto.



O segundo caso de teste foi realizado variando os parâmetros da quantidade de sensores e dados sensoriais, de modo que, foram simulados os envios de dados de dez dispositivos informando temperatura do animal, temperatura ambiente e umidade relativa do ar a cada dez segundos, com um total de 10.800 dados sensoriais por hora. Igualmente ao primeiro teste, nesse caso também foram empregadas informações obtidas de um conjunto dados real. O resultado dessa experimentação pode ser observado nas Figura 27, Figura 28 e Figura 29.

Figura 28 – Primeiro experimento, informações do uso de memória do *driver* LoRa na simulação com doze leituras por minuto.

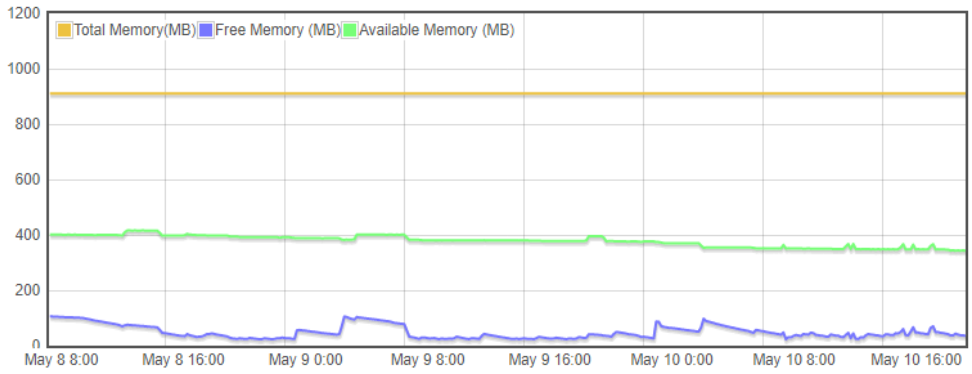


Figura 29 – Primeiro experimento, informações do uso de rede do *driver* LoRa na simulação com doze leituras por minuto.

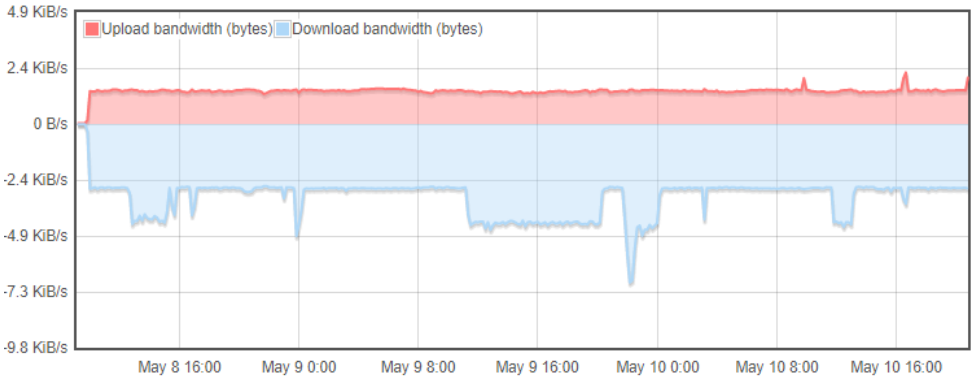
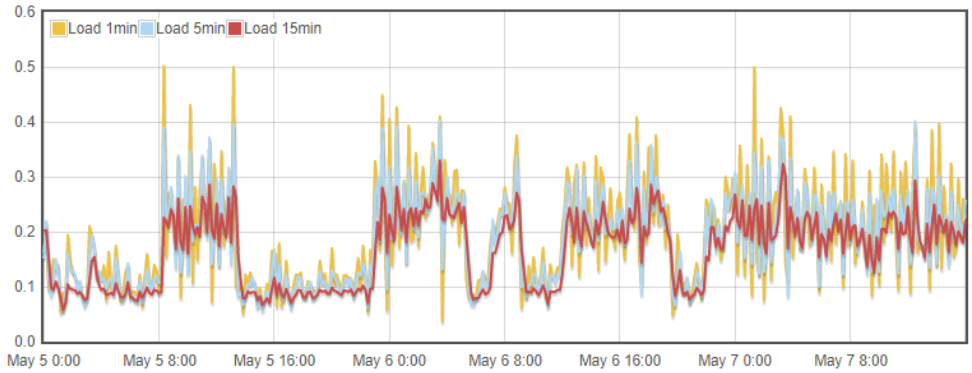


Figura 30 – Segundo experimento, informações do uso de processamento do *driver* LoRa na simulação com 180 leituras por minuto.



Os resultados das experimentações realizadas demonstraram que o *driver* LoRa obteve um bom desempenho, conseguindo tratar as mensagens sem ocasionar sobrecargas ao sistema. De modo geral tanto no primeiro caso de teste quanto no segundo, o uso de CPU ficou abaixo de 50% de um núcleo, a quantidade de memória disponível ficou em torno de 400MB e o uso de rede abaixo de 5kbps. Diferente do *driver* MQTT o *driver* LoRa requisitou uma quantidade maior de memória devido ao uso do ChirpStack e do *daemon* responsável por garantir o funcionamento do *gateway* LoRa.

Figura 31 – Segundo experimento, informações do uso de memória do *driver* LoRa na simulação com 180 leituras por minuto.

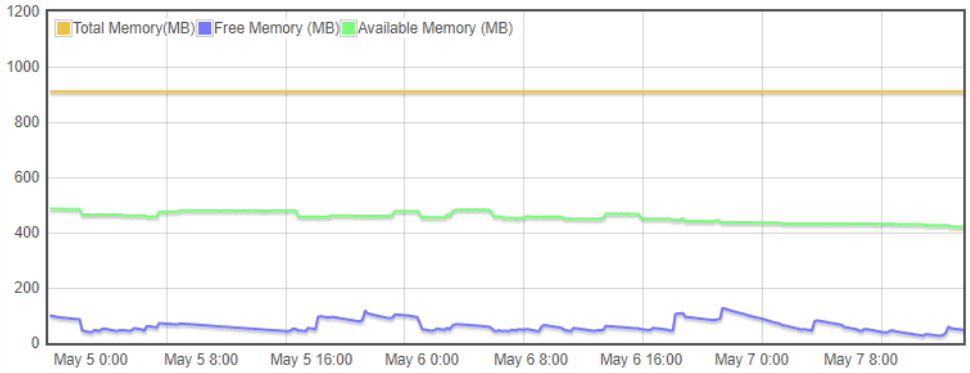
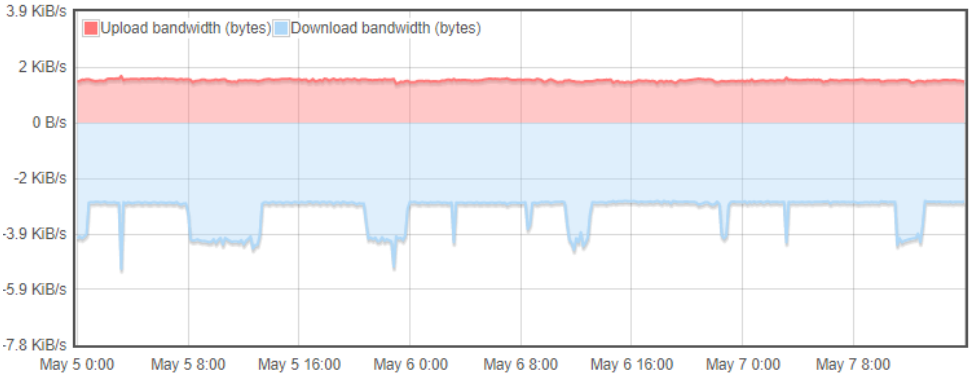


Figura 32 – Segundo experimento, informações do uso de rede do *driver* LoRa na simulação com 180 leituras por minuto.



4.5 Considerações Finais

Sabemos que a pecuária de precisão desempenha um papel de grande importância para contribuição de desenvolvimento sustentável e que fornece ao produtor meios para automatizar as tarefas, auxilia na tomada de decisões e aumenta a granularidade sobre visão do negócio. Em vista disso, temos o e-Cattle uma plataforma IoT com premissa de ser uma ferramenta gratuita, de baixo custo e robusta que atenda desde o produtor familiar a grandes fazendas produtoras de bovinos de corte.

Este trabalho propôs uma extensão as capacidades de comunicação do e-Cattle adicionando dois novos protocolos a lista de suporte da plataforma, sendo o MQTT e o LoRa os escolhidos devido a seus atributos. Essa inclusão ocorreu por meio de desenvolvimento de *Drivers* Dedicados que são disponibilizados para instalação via *snap*.

O *driver* MQTT atua como um tradutor entre os dispositivos sensoriais que falam MQTT e a camada de Coleta de Dados que possui uma API RESTful que fica encarregada de realizar o tratamento dos dados e a sua persistência no banco de dados. O *driver* LoRa atua de maneira semelhante ao *driver* MQTT com a diferença que foi necessário a instalação da aplicação ChirpStack para gerenciar toda a pilha de comunicação pertinentes

ao LoRaWAN.

Os *drivers* passaram por testes para garantir sua robustez e consolidação da sua integração com as outras camadas da plataforma, e os resultados obtidos foram satisfatórios comprovando a viabilidade de uso dos mesmos. O seu código encontra-se disponível na página do projeto da Embrapa Gado de Corte no Github[9] e tem cerca de 120 linhas de código para o *driver* MQTT e 270 linhas de código para o *driver* LoRa.

[9] <https://github.com/e-cattle>

5 Conclusão

A pecuária de precisão é essencial nas atuais circunstâncias em que vivenciamos, onde o cadeia de produção bovina é gradativamente mais competitiva, além de sofrer pressão por outro lado a respeito das questões ambientais cada vez mais frequentes. O uso da tecnologia tem auxiliado a humanidade a superar diversos problemas ao passar dos tempos, e na agropecuária não é diferente.

O e-Cattle surge como uma alternativa de baixo custo, acessível e com pouca exigência de conhecimento para uso, o que o torna ideal para um seguimento onde o público pode variar bastante de poder aquisitivo e erudição. As tecnologias embarcadas na plataforma proveem um ambiente de integração com grande variedade de protocolos, dispositivos e padrões, proporcionando ao usuário uma ampla gama de dispositivos que podem ser utilizados para monitoramento do seu rebanho e propriedade.

O ponto principal desse trabalho é prover meios de comunicação de longa distância que possuam características de baixo consumo energético para utilização propriedades produtoras de bovinos de corte. Com base nessa premissa, foram desenvolvidas duas componentes de *software* que estão integradas ao núcleo da plataforma e-Cattle, chamados de *drivers*. Eles são responsáveis por adicionar novas capacidades de comunicação ao sistema, com protocolos que possuem propriedades inerentes ao cenário abordado. Esses *drivers* tornaram possível a integração de dispositivos que comunicam-se por MQTT e LoRa com o *middleware* BigBoxx por meio da padronização do modo como eles se interagem.

Com o desenvolvimento dos *drivers* dedicados à comunicação IoT, foi possível melhorar o alcance nas comunicações aceitas pela plataforma e-Cattle, tornando-a ainda mais viável para utilização no monitoramento das propriedades criadoras de bovinos de corte.

Por fim, foram realizados diversas simulações para avaliar a capacidade dos *drivers* no tratamento dos dados recebidos pela camada de Coleta de Dados e também o seu impacto na performance do *hardware* do *middleware*. Nesses testes, foram avaliadas questões como: uso de processamento, de rede e de memória, onde não houve demonstração de impacto significativo na performance do equipamento, ou seja, os *drivers* são aptos e capazes de integração à plataforma e-Cattle.

5.1 Trabalhos Futuros

Constantes melhorias e evoluções fazem parte do ciclo de desenvolvimento de qualquer *software* e a plataforma e-Cattle apesar de recente conta com contínuas melhorias, não só dos componentes consolidados assim como incorporação de novos recursos. A seguir são listados algumas sugestões para melhorias em trabalhos futuros:

- **Inclusão de novos protocolos de rede:** atualmente a plataforma e-Cattle conta com suporte aos principais protocolos de comunicação utilizados na IoT. Contudo, a inclusão de outros protocolos como: Sigfox, NBiot, Neul, Zigbee, entre outros, trará uma capacidade maior de agregação de dispositivos e serviços que poderão ser agregados, potencializando a adoção da plataforma.

- **Suporte a outras topologias de rede**: além da topologia estrela que foi adotada como padrão da plataforma, é relevante promover o suporte a outros modelos como por exemplo as redes híbridas, podendo ajudar solucionar problemas de alcance entre os dispositivos e de viabilizar a inclusão de outras redes existentes que não partilham da mesma estrutura.

- **Integração do protocolo de comunicação CoAP:** assim como o MQTT, o CoAP (*Constrained Application Protocol*) é um protocolo destinado para comunicação M2M e IoT com foco em dispositivos de *hardware* simples. Esse protocolo possui características muito semelhantes ao HTTP e pode ser utilizado como uma alternativa extramente viável.

- **Testes em ambiente real com LoRa:** durante o desenvolvimento do *driver* para comunicação com dispositivos LoRa não foi possível realizar testes com diversos *endpoints* devido à limitação na quantidade de equipamentos disponíveis. Todavia, seria relevante a execução de experimentação com múltiplos dispositivos de diferentes fabricantes para avaliar o comportamento do sistema.

Referências

ABIEC. Beef Report 2020. [S.l.], 2020. Disponível em: <http://abiec.com.br/publicacoes/beef-report-2020/>. Acesso em: 29 mai 2020. Citado na página 17.

ANTON-HARO, C.; DOHLER, M. Machine-to-machine (M2M) communications: architecture, performance and applications. [S.l.]: Elsevier, 2014. Citado na página 26.

ASHTON, K. That 'internet of things' thing. RFiD Journal, jun 2009. Disponível em: <http://www.rfidjournal.com/articles/view?4986>. Citado na página 18.

BUCCI, G.; BENTIVOGLIO, D.; FINCO, A. Precision agriculture as a driver for sustainable farming systems: State of art in literature and research. Quality - Access to Success, v. 19, p. 114–121, 03 2018. Citado na página 17.

CARROMEU, C. e-Cattle: Uma Plataforma de IoT para Pecuária de Precisão. Tese (Doutorado) — Universidade Federal de Mato Grosso do Sul, nov 2019. Citado 9 vezes nas páginas 20, 36, 38, 39, 40, 41, 42, 47 e 54.

CáCERES, B. de A. Barramento de serviços para consulta de dados sensoriais em IoT na Plataforma e-Cattle. Dissertação (Mestrado) — Universidade Federal de Mato Grosso do Sul - UFMS, nov 2020. Citado na página 41.

David, V. et al. Iot based automated indoor agriculture system using node-red and ibm bluemix. In: 2021 6th International Conference on Inventive Computation Technologies (ICICT). [s.n.], 2021. p. 157–162. Disponível em: <https://ieeexplore.ieee.org/document/9358672>. Citado 3 vezes nas páginas 34, 37 e 70.

DYBA, T.; DINGSOYR, T.; HANSSEN, G. K. Applying systematic reviews to diverse study types: An experience report. In: IEEE. First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007). [S.l.], 2007. p. 225–234. Citado na página 68.

EVANS, D. The internet of things: How the next evolution of the internet is changing everything. CISCO white paper, v. 1, n. 2011, p. 1–11, 2011. Citado na página 18.

FERDOUSH, T. E.; TAHSIN, M.; TAHER, K. A. Innovative smart farming system with wimax and solar energy. In: Proceedings of the International Conference on Computing Advancements. New York, NY, USA: Association for Computing Machinery, 2020. (ICCA 2020). ISBN 9781450377782. Disponível em: <https://doi.org/10.1145/3377049.3377063>. Citado 3 vezes nas páginas 32, 37 e 70.

FERNANDEZ-AHUMADA, L. M. et al. Proposal for the Design of Monitoring and Operating Irrigation Networks Based on IoT, Cloud Computing and Free Hardware Technologies. SENSORS, 19, n. 10, MAY 2 2019. ISSN 1424-8220. Citado 5 vezes nas páginas 9, 35, 36, 37 e 70.

GERMANI, L. et al. An IoT Architecture for Continuous Livestock Monitoring Using LoRa LPWAN. ELECTRONICS, 8, n. 12, DEC 2019. Citado 3 vezes nas páginas 34, 37 e 70.

GLAROUDIS, D.; IOSSIFIDES, A.; CHATZIMISIOS, P. Survey, comparison and research challenges of iot application protocols for smart farming. Computer Networks, Elsevier, v. 168, p. 107037, 2020. Citado 2 vezes nas páginas 27 e 28.

GUBBI, J. et al. Internet of things (iot): A vision, architectural elements, and future directions. Future generation computer systems, Elsevier, v. 29, n. 7, p. 1645–1660, 2013. Citado na página 68.

IBM. MQTT V3.1 Protocol Specification. 2012. Disponível em: <http://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html>. Acesso em: 01 dez. 2019. Citado na página 27.

Islam, A. et al. Iot based power efficient agro field monitoring and irrigation control system : An empirical implementation in precision agriculture. In: 2018 International Conference on Innovations in Science, Engineering and Technology (ICISET). [s.n.], 2018. p. 372–377. Disponível em: <https://ieeexplore.ieee.org/document/8745605>. Citado 3 vezes nas páginas 33, 37 e 70.

JAYAGEETHA, J. et al. Smart monitoring and management system for efficient cultivation. In: IEEE. 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS). 2020. p. 421–425. Disponível em: <https://ieeexplore.ieee.org/document/9074219>. Citado 3 vezes nas páginas 33, 37 e 70.

Jeaunita, T. C. J. et al. An automated greenhouse system using agricultural internet of things for better crop yield. In: Smart Cities Symposium 2018. [s.n.], 2018. p. 1–6. Disponível em: <https://ieeexplore.ieee.org/document/8643194>. Citado 3 vezes nas páginas 32, 37 e 70.

Joshitha, C. et al. Lorawan based cattle monitoring smart system. In: 2021 7th International Conference on Electrical Energy Systems (ICEES). [s.n.], 2021. p. 548–552. Disponível em: <https://ieeexplore.ieee.org/document/9383749/>. Citado 3 vezes nas páginas 34, 37 e 70.

KITCHENHAM, B. A.; DYBA, T.; JORGENSEN, M. Evidence-based software engineering. In: IEEE. Proceedings. 26th International Conference on Software Engineering. [S.l.], 2004. p. 273–281. Citado na página 67.

Kodali, R. K.; Kuthada, M. S.; Yogi Borra, Y. K. Lora based smart irrigation system. In: 2018 4th International Conference on Computing Communication and Automation (ICCCA). [s.n.], 2018. p. 1–5. Disponível em: <https://ieeexplore.ieee.org/document/8777583>. Citado 3 vezes nas páginas 33, 37 e 70.

LEE, I.; LEE, K. The internet of things (IoT): Applications, investments, and challenges for enterprises. Business Horizons, v. 58, n. 4, p. 431 – 440, 2015. ISSN 0007-6813. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0007681315000373>. Citado na página 18.

LEFTRONIC. IoT statistics and trends to know in 2021. 2021. Disponível em: <https://leftronic.com/blog/internet-of-things-statistics/>. Acesso em: 30 apr 2021. Citado na página 18.

LORA ALLIANCE. Lora alliance® annual report. 2020. Disponível em: <https://lora-alliance.org/about-lora-alliance/>. Acesso em: 15 feb 2020. Citado 3 vezes nas páginas 24, 25 e 26.

MIKELSTEN, D. Automação e Tecnologias Emergentes. Cambridge Stanford Books, 2009. (Inteligência Artificial: A Quarta Revolução Industrial). Disponível em: <https://books.google.com.br/books?id=WR7NDwAAQBAJ>. Citado na página 67.

MILLER, B.; ATKINSON, R. D. Raising european productivity growth through ict. ITIF, June, 2014. Citado na página 17.

MINH, Q. T. et al. A cost-effective smart farming system with knowledge base. In: Proceedings of the Eighth International Symposium on Information and Communication Technology. New York, NY, USA: Association for Computing Machinery, 2017. (SoICT 2017), p. 309–316. ISBN 9781450353281. Disponível em: <https://doi.org/10.1145/3155133.3155151>. Citado 5 vezes nas páginas 9, 31, 32, 37 e 70.

MISHRA, D. et al. Smart agriculture system using iot. In: Proceedings of the Third International Conference on Advanced Informatics for Computing Research. New York, NY, USA: Association for Computing Machinery, 2019. (ICAICR '19). ISBN 9781450366526. Disponível em: <https://doi.org/10.1145/3339311.3339350>. Citado 3 vezes nas páginas 31, 37 e 70.

MUANGPRATHUB, J. et al. IoT and agriculture data analysis for smart farm. Computers and Electronics in Agriculture, v. 156, p. 467 – 474, 2019. ISSN 0168-1699. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0168169918308913>. Citado 5 vezes nas páginas 9, 34, 35, 37 e 70.

RUBIO-APARICIO, J. et al. Design and implementation of a mixed IoT LPWAN Network Architecture. Sensors, v. 19, p. 675, 02 2019. Citado 3 vezes nas páginas 19, 23 e 24.

SANTOS, V. B. dos. Desenvolvimento de software para plataforma de internet das coisas aplicada à pecuária de precisão. Dissertação (Mestrado) — Universidade Federal de Mato Grosso do Sul - UFMS, mai 2018. Citado 3 vezes nas páginas 40, 41 e 42.

SHAMSHIRI, R. et al. Research and development in agricultural robotics: A perspective of digital farming. International Journal of Agricultural and Biological Engineering, v. 11, p. 1–14, 07 2018. Citado na página 17.

SILVA, L. B. da. Modelagem e Persistência de Dados Sensoriais na Plataforma e-Cattle. Dissertação (Mestrado) — Universidade Federal de Mato Grosso do Sul - UFMS, set 2018. Citado 3 vezes nas páginas 40, 41 e 72.

TECHPLAYON. Low power wide area networks (LPWAN). 2017. Disponível em: <http://www.techplayon.com/low-power-wide-area-networks-lpwan/>. Acesso em: 21 jan 2020. Citado na página 19.

TREVISAN, R. Dicionário michaelis. Editora Melhoramentos Ltda, 2015. Citado na página 68.

YOKOTANI, T.; SASAKI, Y. Comparison with HTTP and MQTT on required network resources for IoT. In: IEEE. 2016 international conference on control, electronics, renewable energy and communications (ICCEREC). [S.l.], 2016. p. 1–6. Citado na página 27.

Yoon, C. et al. Implement smart farm with iot technology. In: 2018 20th International Conference on Advanced Communication Technology (ICACT). [s.n.], 2018. p. 749–752. Disponível em: <https://ieeexplore.ieee.org/document/8323908>. Citado 3 vezes nas páginas 32, 37 e 70.

YUAN, M. Getting to know MQTT. may 2017. Disponível em: <https://developer.ibm.com/articles/iot-mqtt-why-good-for-iot/>. Citado na página 26.

Apêndices

APÊNDICE A – Revisão Sistemática

Uma revisão sistemática foi realizada com finalidade de buscar e selecionar pesquisas relacionadas ao tema proposto, baseados nas palavras-chaves definidas deste trabalho. Nesta seção é detalhado o levantamento das propriedades pertinentes à revisão sistemática, que segundo Kitchenham, Dyba e Jorgensen (2004) pode ser dividido em quatro etapas: Planejamento, Condução, Extração dos Dados e Análise dos Resultados.

A.1 Planejamento

Visando encontrar o estado da arte à respeito de pecuária de precisão, foi realizado busca que formou uma base quantitativa. Em seguida foram ponderadas a amplitude e a proximidade dos resultados obtidos para compor a base qualitativa.

Auxiliar os pesquisadores a encontrar estudos relevantes é a principal razão para formulação de questões de pesquisa (QP) (KITCHENHAM; DYBA; JORGENSEN, 2004). Portanto, a busca pela maior quantidade de resultados possíveis associado a obtenção de indicativos sobre os estudos existentes torna essencial a definição dos objetivos da pesquisa. Logo, foram definidas as questões de pesquisas listadas abaixo:

1. Quais trabalhos apresentam uma proposta de automação no domínio da pecuária de precisão?
2. Entre esses trabalhos, quais fazem uso de sensores que utilizam protocolos de comunicação comuns IoT para auxílio na tomada de decisão do manejo automatizado?
3. Quais possuem uma arquitetura IoT baseada em camadas e barramentos de serviços?

A construção das *strings* de busca foram definidas de acordo com o contexto do trabalho: pecuária, automação, Internet das Coisas e sensores. Foram definidas *strings* genéricas de buscas que puderam ser ajustadas conforme a base pesquisada.

Podem ser vistas na Tabela 4 as *strings* de busca utilizadas como conceitos-chave para as áreas pesquisadas:

1. **Automação:** É a tecnologia pela qual um processo ou procedimento é realizado com o mínimo de assistência humana (MIKELSTEN, 2009).
2. **Internet das Coisas:** É uma rede composta por sensores que oferece a capacidade de medir, inferir e entender indicadores ambientais e recursos naturais a ambientes

Tabela 4 – Termos relevantes para construção de *strings* padrão (sinônimos).

Automação	**Internet das Coisas**	**Pecuária**
AUTOMATION	LPWAN	LIVESTOCK
AUTOMATIZATION	M2M	CATTLE
	LoRa	AGRICULTURE
	MQTT	FARM

urbanos; e as informações são compartilhadas entre plataformas (GUBBI et al., 2013).

3. **Pecuária:** Conjunto de atividades que envolve a criação e o tratamento do gado (TREVISAN, 2015).

Uma *string* de busca foi produzida a partir dos conceitos-chaves listados na Tabela 4 e derivou na seguinte *query* de busca:

```
(AUTOMATION or AUTOMATIZATION) and
(LPWAN or M2M or LoRa or MQTT) and
(LIVESTOCK or CATTLE or AGRICULTURE or FARM)
```

Para seleção das bases de pesquisas bibliográficas que fundamentaram o trabalho, foram utilizadas as principais da área de computação levantadas por Dyba, Dingsoyr e Hanssen (2007), que são: ACM (*ACM Digital Library*), IEEE (*IEEE Xplore Digital Library*), *Web of Science* e *Science Direct / Elsevier*. Além dessas, foram incluídas as Scopus por sua grande abrangência de conteúdos internacionais e a Scielo que possui grande parte do seu conteúdo produzido em países latino-americanos. Ambas possuem uma grande quantidade de materiais na área de computação de deveras importância.

As pesquisas escolhidas foram selecionadas com base na relevância do tema para o trabalho, com conteúdo atualizado e de qualidade. Na Tabela 5 são listadas as bases informadas anteriormente.

Tabela 5 – Bases de buscas empregues.

Base de pesquisa	**Acrônimo**	**Endereço**
ACM Digital Library	ACM	<https://dl.acm.org/>
IEEE Xplore Digital Library	IEEE	<https://ieeexplore.ieee.org/>
ScienceDirect / Elsevier	SCD	<https://www.sciencedirect.com/>
Web of Science	WOS	<https://apps.webofknowledge.com/>
Scopus	SCP	<https://www.scopus.com/>
Scielo	SCL	<https://www.scielo.org/>

Por fim, os resultados obtidos nas buscas passaram por uma etapa de refinamento onde eles são incluídos pelo Critério de Inclusão (CI) ou excluídos pelo Critério de Exclusão (CE). A Tabela 6 apresenta as condições avaliadas no CI e a Tabela 7 mostra os requisitos avaliados no CE.

Tabela 6 – Critérios de Inclusão (CI) de busca do mapeamento sistemático.

Item	**Critério adotado**
CI01	Estudos que apresentem coleta de dados por sensores na agropecuária.
CI02	Estudos que apresentem automação na agropecuária.
CI03	Estudos baseados no uso de protocolos de comunicação IoT.

Tabela 7 – Critérios de Exclusão (CE) de busca do mapeamento sistemático.

Item	**Critério adotado**
CE01	Estudo que não atenda aos critérios de inclusão.
CE02	Estudo publicado anteriormente à 2017.
CE03	Estudo que não esteja em inglês ou em português.
CE04	Estudo incompleto, indisponível ou similar a outros jé elencados.
CE05	Estudo não publicado em anais de eventos, em revistas, como relatórios técnicos, capítulos de livros, teses ou dissertações.

A.2 Condução

A etapa de condução das buscas foi composto pelos itens elencados das bases de dados informadas na Tabela 5 juntamente com as *strings* da Tabela 4. Realizado a identificação dos trabalhos, os mesmos passaram por um processo de seleção. A seleção ocorreu aplicando filtros definidos com base nos critérios definidos na Tabela 7.

Tabela 8 – Síntese dos resultados da pesquisa e aplicação dos critérios de exclusão.

Base de Dados	**Resultados**	**CE01**	**CE02**	**CE03**	**CE04**	**CE05**	**Total**
ACM	75	−49	−11	0	0	−12	3
IEEE	28	−20	−1	0	0	0	7
SCD	349	−150	−74	0	0	−124	1
WOS	11	−5	−3	0	0	−1	2
SCP	27	−21	−2	0	0	−4	0
SCL	1	0	0	−1	0	0	0
Total	491	−245	−91	−1	0	−141	13

A.3 Extração dos Dados

Selecionado os estudos de maior relevância à área de pesquisa desse trabalho (Tabela 8), o passo seguinte consiste na extração das informações dos mesmos. O resultado desse levantamento pode ser observado na Tabela 9.

Tabela 9 – Estudos resultantes da seleção de dados.

Identificador	**Título**	**Ano**
ACM01	A Cost-Effective Smart Farming System with Knowledge Base (MINH et al., 2017)	2017
ACM02	Smart Agriculture System Using IoT (MISHRA et al., 2019)	2019
ACM03	Innovative Smart Farming System with WiMAX and Solar Energy (FERDOUSH; TAHSIN; TAHER, 2020)	2020
IEEE01	Implement smart farm with IoT technology (Yoon et al., 2018)	2018
IEEE02	An automated greenhouse system using agricultural Internet of Things for better crop yield (Jeaunita et al., 2018)	2018
IEEE03	LoRa Based Smart Irrigation System (Kodali; Kuthada; Yogi Borra, 2018)	2018
IEEE04	Smart Monitoring and Management System for Efficient Cultivation (JAYAGEETHA et al., 2020)	2020
IEEE05	IoT Based Power Efficient Agro Field Monitoring and Irrigation Control System : An Empirical Implementation in Precision Agriculture (Islam et al., 2018)	2018
IEEE06	LoRaWAN based Cattle Monitoring Smart System (Joshitha et al., 2021)	2021
IEEE07	IoT based Automated Indoor Agriculture System Using Node-RED and IBM Bluemix (David et al., 2021)	2021
SCD01	IoT and Agriculture Data Analysis for Smart Farm (MUANGPRATHUB et al., 2019)	2019
WOS01	An IoT architecture for continuous livestock monitoring using lora LPWAN (GERMANI et al., 2019)	2019
WOS02	Proposal for the Design of Monitoring and Operating Irrigation Networks Based on IoT, Cloud Computing and Free Hardware Technologies (FERNANDEZ-AHUMADA et al., 2019)	2019

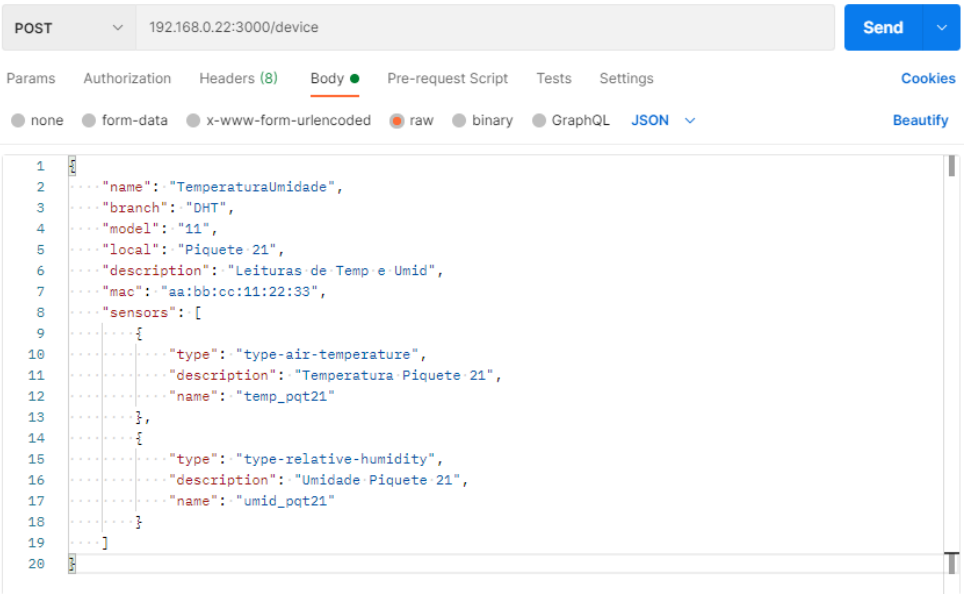
A.4 Análise dos Resultados

A análise dos resultados obtidos por meio dessa revisão sistemática encontram-se detalhados no Capítulo 3 e serviram como fundamentos para desenvolvimento deste trabalho.

APÊNDICE B – Modelo de Contrato e Entidades do Banco de Dados Mongo

Ao inserir um novo dispositivo na plataforma e-Cattle, o mesmo deve informar o seu perfil com as suas propriedades ao *BigBoxx*. Esse anúncio deve ocorrer por meio de uma mensagem HTTP POST no *socket* TCP http://<IP-BigBoxx>:3000/device contendo todas informações relativas ao dispositivo e os tipos de sensores que ele contém. O conteúdo dessa mensagem deve estar em formato JSON e ao seu conteúdo dá-se o nome de "contrato", um exemplo pode ser visto na Figura 33. O contrato contém informações como: *name* um nome para o dispositivo, *branch* qual a sua marca, *model* qual seu modelo, *local* o local em que está instalado, *description* uma descrição, *mac* o endereço *mac-address* da interface de rede utilizada para comunicação com o BigBoxx, uma lista contendo um ou mais sensores com informações de *type* informa o tipo de dado daquele sensor (baseado no modelo definido no banco de dados Tabela 10), *description* uma descrição e *name* um nome. Se o envio do contrato for enviado e aceito com sucesso, uma mensagem de retorno é recebida, conforme Figura 34.

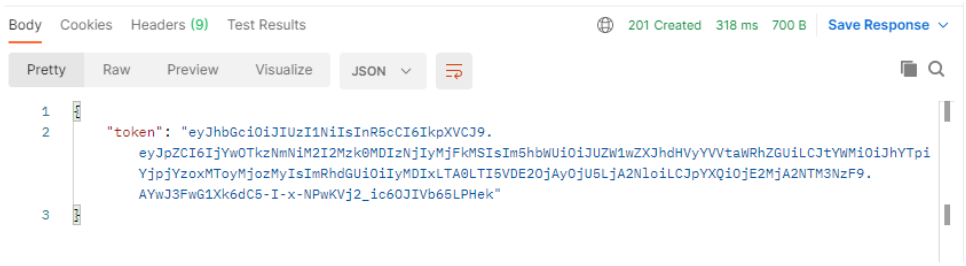
Figura 33 – Envio de um contrato por um dispositivo contendo sensores de temperatura e umidade relativa do ar.



A camada de persistência é encarregada de armazenar os dados em um banco de dados do tipo não-relacional (NoSQL), que foi definido o MongoDB[1]. Esse tipo de banco de dados é indicado em casos que existe um grande volume e pluralidade de dados e sua

[1] <https://www.mongodb.com/>

Figura 34 – Resposta obtida após envio de um contrato.



abordagem tem maior facilidade no armazenamento na forma objeto chave-valor (SILVA, 2018).

Os dados sensoriais são classificados em quatro grupos de variáveis: Comportamental, Contextual, Fisiológico ou Microclima. Os grupos Comportamental e Fisiológico estão relacionados com informações pertinentes ao animal, enquanto o grupo Contextual e Microclima são concernentes ao local. As variáveis são conhecidas como Entidades no banco de dados e os tipos existentes podem ser vistos na Tabela 10. Cada entidade possui atributos próprios e representa uma informação coletada pelos sensores, sendo que os atributos são características de cada entidade e possuem uma chave, valor, unidade de medida e limites válidos (SILVA, 2018).

Tabela 10 – Entidades existentes no e-Cattle.

Comportamental	**Contextual**	**Fisiológico**	**Microclima**
Accelerometer	GateOpened	AnimalWeight	AirTemperature
AnimalSpeed	GeographicCoordinate	BodyTemperature	BlackGlobeTemperature
Gyroscope		HeartRate	CH4
Magnetometer		RespiratoryFrequency	CO2
		RetalTemperature	DewPointTemperature
			DryBulbTemperature
			PH
			Precipitation
			RelativeHumidity
			SoilTemperature
			SoilMoisture
			SoilWaterPotencial
			SoilNitrogen
			SolarRadiation
			WaterTemperature
			WetBulbTemperature
			WindSpeed

- **Comportamental**: possui a entidade *AnimalSpeed*, que registra a velocidade do animal em metros por segundo.
- **Contextual**: possui a entidade *GateOpened*, representa o estado de um portão se está aberto ou fechado.

- **Fisiológico**: possui a entidade *AnimalWeight*, que contém o peso do animal em quilograma.
- **Microclima**: possui a entidade *RelativeHumidity*, refere-se a umidade relativa do ar em porcentagem.

Cada tipo de dado sensorial possui um *Schema* que o define no banco de dados, na Figura 35 é possível observar as características adotadas para o tipo de dados umidade relativa do ar. O modelo possui o atributo *value* que aceita valores positivos em ponto flutuante com até duas casas de precisão (variando entre 0 e 100), o campo *resource* guarda informação de onde esse dado foi coletado e por fim os campos *date* e *storaged* que representam respectivamente, a data de leitura e a data de armazenamento do dado no banco de dados.

Figura 35 – *Schema* que define as características da Entidade Microclima *Relative Humidity*.

```
'use strict'

const mongoose = require('mongoose')
const Schema = mongoose.Schema

const relativeHumidity = new Schema({
  device: {
    type: Schema.Types.ObjectId,
    ref: 'Device',
    required: true
  },
  value: {
    type: Number,
    min: 0,
    max: 100,
    validate: /^\d{0,3}(\.\d{1,2})?$/,
    required: true
  },
  date: {
    type: Date,
    required: true
  },
  resource: {
    type: String,
    required: false
  },
  storaged: {
    type: Date,
    default: Date.now()
  }
}, { collection: 'type-relative-humidity' })

module.exports = mongoose.model('type-relative-humidity', relativeHumidity)
```