Sistema de Comunicação LoRa: Uma solução para Transmissão de Dados no Agronegócio envolvendo Longas Distâncias e Baixo Custo

Diego F. Tozetto*, Rodrigo V. Silva*, Roberto Z. Freire*

* Programa de Pós-Graduação em Engenharia de Produção e Sistemas – PPGEPS, Escola Politécnica – EP,
Pontifícia Universidade Católica do Paraná – PUCPR,
Imaculada Conceição, 1155, CEP. 80215-901
(e-mails: diegotozetto@gmail.com, rodrigo.viti@gmail.com, roberto.freire@pucpr.br)

Abstract: The Brazilian territory covers about 5,073,324 rural establishments, of which 90% have less than 100 hectares. Which leads to concentrated production in terms of income, with less than 1% of farms contributing half of the total production value in 2017. For a long time, it was believed that the cause of this concentration was linked to uneven land distribution. However, in more recent studies it was realized that access to technology is the main reason for the concentration of production. In the present context, this work aims to use LoRa (Long Range) radiofrequency technology, which allows communication over long distances, with low power and reduced cost, to facilitate the monitoring of activities generated in agriculture along with a farm, where the technology and the internet signal are precarious. The proposed system consists of the development of hardware, firmware, and software for a local server and LoRa gateway, which receives information from different devices and sends it to the server. The solution was able to communicate of 58 bytes over 2 Km with an average of 93% received packets, in a flat and open terrain, where the transmission has a more straight view between the gateway and the devices. However, mountainous terrains prove to be difficult to propagate LoRa modulation, resulting in loss of signal and, consequently, of packets.

Resumo: O território brasileiro abrange cerca de 5.073.324 de estabelecimentos rurais, dos quais 90% possuem menos de 100 hectares. O que leva a uma produção concentrada em termos de renda, sendo que menos de 1% das fazendas contribuíram com metade do valor total da produção em 2017. Durante muito tempo, acreditou-se que a causa desta concentração estava ligada a distribuição desigual de terras. Contudo, em estudos mais recentes percebeu-se que o acesso à tecnologia é o principal motivo de concentração da produção. No presente contexto, este trabalho tem como finalidade utilizar a tecnologia de radiofrequência LoRa (*Long Range*), que permite a comunicação em longas distâncias, com baixa potência e custo reduzido, para facilitar o monitoramento das atividades geradas no agronegócio ao longo de uma fazenda, onde a tecnologia e o sinal de internet são precários. O sistema proposto consiste no desenvolvimento de hardware, firmware e software para um servidor local e *gateway* LoRa, que recebe as informações de diferentes dispositivos, enviando-os para o servidor. A solução foi capaz de comunicar pacotes de 58 *bytes* a uma distância de 2 Km, com uma média de 93% pacotes recebidos, em terrenos planos e descampados onde a transmissão possui uma visão mais direta entre o *gateway* e os dispositivos. Contudo, terrenos montanhosos demonstram-se de difícil propagação da modulação LoRa, resultando em perda de sinal e consequentemente, de pacotes.

Keywords: Agribusiness; Gateway; IoT; LoRa; Server.

Palavras-chaves: Agronegócio; *Gateway*; IoT; LoRa; Servidor.

1. INTRODUÇÃO

O Brasil possui cerca de 41% do seu território representado por estabelecimentos agropecuários (IBGE, 2019). De acordo com relatórios recentes (CEPEA, 2021), pode-se afirmar que o PIB do agronegócio em 2020 no Brasil teve um aumento de 24,31% representando uma participação de 26,6% no PIB brasileiro, chegando a quase R$ 2 trilhões.

Conforme discutido por Souza, Gomes e Alves (2020), a agricultura brasileira é extremamente concentrada em termos de renda, sendo que menos de 1% das fazendas foram responsáveis por cerca de 50% do valor total da produção em 2017. De acordo com o mesmo estudo, o acesso à tecnologia é a principal causa da concentração da produção.

Segundo dados do Censo Agropecuário de 2017 (IBGE, 2019), apenas 9,5% da população agropecuária possui acesso à internet, mesmo com o avanço constante da tecnologia e a

necessidade de uma produtividade maior, conectar dispositivos eletrônicos podem se tornar um desafio. Isso em razão de grandes distâncias e baixa conectividade à internet. Nesse caso, um sistema que possibilite o agrupamento de dados de diversos sensores e dispositivos eletrônicos relacionados à Internet das Coisas (*Internet of Things* - IoT) utilizando comunicação LoRa, pode ser de grande ajuda para a inclusão tecnológica da parcela menos favorecida do agronegócio.

Novas tecnologias podem trazer benefícios as pessoas, por exemplo, a Internet das Coisas já vem exercendo um papel importante no agronegócio, alguns sistemas disponíveis baseados nesta tecnologia visam a integração de sensores voltados à agricultura com um servidor web através da comunicação LoRa (HEBLE et al., 2018; MA; CHEN, 2018; JI et al., 2019) e transmissão de imagem (YOON et al., 2019).

Neste trabalho é proposto um sistema para realizar a comunicação entre um servidor local e dispositivos IoT conectados através do *gateway* utilizando tecnologia LoRa. O trabalho ainda apresenta a avaliação de desempenho do sistema em termos de alcance de comunicação.

Este artigo é apresentado em 5 partes: embasamento teórico, sistema proposto, experimentos, resultados e conclusões. Em conceitos teóricos (seção 2) é apresentado um referencial teórico para o entendimento da tecnologia LoRa. A seção 3 descreve o sistema desenvolvido que é utilizado no processo de comunicação. A seção 4 apresenta os experimentos realizados em campo, enquanto a seção 5 demonstra os resultados obtidos. Finalmente, a seção 6 apresenta a conclusão deste estudo, as restrições do projeto e os desdobramentos para trabalhos futuros.

2. EMBASAMENTO TEÓRICO

As próximas subseções têm por objetivo trazer os conceitos das tecnologias adotadas neste trabalho.

2.1 LoRa

LoRa (*Long Range*) é uma tecnologia proprietária de radiofrequência, criada pela Semtech Corporation, e promovida pela associação LoRa Alliance. Uma relação ampla e sem benefícios lucrativos, no qual existe uma sociedade entre grandes empresas espalhadas pelo mundo, que tem como anseio a evolução e aplicação da tecnologia LoRa na Internet das Coisas (IoT). Apareceu como solução para as redes LPWANs (*Low-Power Wide-Area Network*), com o propósito de atingir aplicações com baixo consumo de energia e taxa de transmissão, bem como as que exigem longas distâncias (AUGUSTIN, 2016).

De acordo com Bankov et al. (2016), a técnica de modulação do LoRa se baseia na técnica de codificação CSS (*Chirp Spread Spectrum*), inicialmente desenvolvida para aplicações militares e de radar. A técnica do CSS consiste no uso de pulsos modulados em frequência linear de banda larga cuja frequência aumenta ou diminui com base na informação codificada, por isso com baixas taxas de transmissão e alta largura de banda consegue-se um grande alcance por meio deste tipo de modulação.

Yoon et al. (2019) afirmam em seu estudo, que a camada física do LoRa faz uso de um certo número de parâmetros dos quais podem ser configurados em um conjunto diferente de muitas combinações, para que seja possível assegurar uma boa qualidade de comunicação e menos consumo de energia. Os mesmos autores também afirmam que os parâmetros mais importantes da camada física do LoRa são: frequência da portadora, fator de espalhamento (*Spreading Factor* - SF), largura de banda (*BandWidth* - BW), potência de transmissão e taxa de código (*Code Rate* - CR). A combinação dos parâmetros permite uma taxa de transmissão entre 50 bps e 37 kbps. Os autores também descrevem que o LoRa opera nas seguintes bandas de frequência: 915 MHz nos Estados Unidos da América, 866 MHz na Europa e 433 Mhz na Ásia.

No Brasil, a frequência de operação para as faixas ISM (*Industrial, Scientific e Medical*) é regulamentada pela Agência Nacional de Telecomunicações (ANATEL). Essa regulamentação acontece por meio de alguns atos e resoluções, entre as mais relevantes está a resolução nº 454, de 1 de dezembro de 2006, que regulamenta a frequência de operação para o LoRa entre 902MHz e 928MHz, com uma frequência típica de 915MHz.

2.2 Arquitetura de Software

De acordo com Santos (2015), o HTTP (*Hypertext Transfer Protocol*) é um protocolo usado para transferir dados pela web, responsável por um grande tráfego de dados de maneira eficiente e rápida entre um servidor e um cliente. Segundo o autor, o HTTP suporta diversos tipos de métodos, também conhecidos como verbos, que indicam qual comportamento o servidor deve tomar. Por isso, a cada nova requisição é mandatório que o cliente defina um método, entre os existentes (GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, TRACE e PATCH), conforme a semântica desejada.

Logo, surgiu um novo conceito conhecido como REST (*Representational State Transfer*), que define um conjunto de regras que garantem a elaboração de um sistema com interfaces bem definidas, escalável, extensível e tolerante a falhas. Isso porque, anteriormente a sua criação costumava-se utilizar somente os métodos GET e POST e a maneira REST garantiu que fosse empregado todos os métodos criados (RODRIGUES, 2009).

Além disso, segundo Dias (2016), tem-se o conceito de API (*Application Programming Interface*) que estabelece um conjunto de definições e protocolos que garantem a comunicação de dados entre aplicações, empregando requisições HTTP. Portanto, ter uma REST API, significa usufruir dos padrões definidos pelo estilo de arquitetura REST junto a uma API para acessar o servidor.

2.3 Frameworks

A seguir apresenta-se os conceitos dos dois frameworks utilizados neste trabalho, são eles: Flask e FreeRTOS.

Flask é um micro framework projetado para utilizar a linguagem Python para elaborar aplicativos web. Para isso, emprega a especificação WSGI (*Web Server Gateway Interface*), que define uma interface simples e universal para os servidores e aplicativos web (THE PALLETS PROJECTS, 2019). Este framework é bastante utilizado por facilitar e acelerar os primeiros passos de desenvolvimento, bem como por permitir que se expanda para aplicativos cada vez mais complexos. Além disso, possui diversas bibliotecas que tornam a ferramenta mais poderosa, tal como o Jinja2, que facilita a inserção de *templates* HTML dentro das aplicações Python (OLIVEIRA, 2019).

Segundo Melot (2019), O FreeRTOS é um framework de tempo real projetado para rodar em microcontroladores. Uma vez que, os microcontroladores são limitados por recursos, qualquer sistema operacional a ser executado nele precisa ser adequadamente projetado. Um sistema operacional de tempo real é construído para ter um padrão de execução determinístico. Com isso, a execução em tempo real significa cumprir todos os prazos definidos, sem necessariamente executar ela rapidamente. Por isso, pode-se ter paralelismo, sem a necessidade de executar cada tarefa sequencialmente, garantindo controle sobre as tarefas, uma vez que o fluxo de execução acontece de acordo com as prioridades.

3. SISTEMA DE TRANSMISSÃO DE DADOS

O sistema de transmissão de dados proposto neste estudo, destacado na Fig. 1, tem como finalidade utilizar a tecnologia LoRa para facilitar o monitoramento das atividades geradas no agronegócio ao longo de uma fazenda, onde o sinal de internet é precário. Por esse motivo, o trabalho em questão apresenta o desenvolvimento de hardware, firmware e software para um *gateway* LoRa e servidor local.

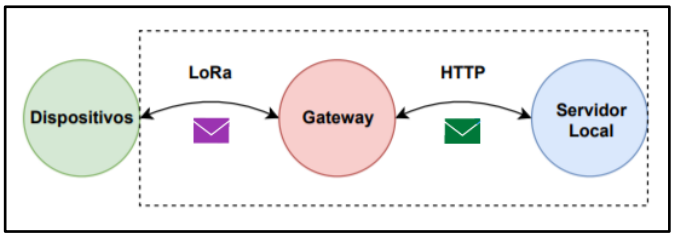

Fig. 1. Diagrama de transmissão de dados do sistema.

Nas próximas subseções apresentam-se os componentes de hardware, as configurações do transceptor, mostram-se os diferentes tipos de comunicação encontrados no sistema, o formato do protocolo, o funcionamento do firmware e do software, a estrutura do banco de dados e a mecânica do projeto.

3.1 Hardware

O hardware é composto por um microcontrolador, um módulo transceptor, uma antena SMA (*SubMiniature version A*) de 6 dBi, um módulo de cartão micro SD para armazenamento local e uma fonte de alimentação externa, com tensão entre 9 V e 18 V que é regulada internamente para 3.3 V e 5 V. A Fig. 2 contém o bloco *gateway* que representa o diagrama de módulos do hardware.

Considera-se importante que, para este tipo de aplicação, o microcontrolador possua boa capacidade de processamento e um módulo de comunicação sem fio (Wi-fi) integrado. Neste caso, escolheu-se o NodeMCU-32S. Também, optou-se pelo E32-915T30D, da Ebyte, para compor o módulo transceptor, uma vez que contém internamente um ajuste da comunicação SPI (*Serial Peripheral Interface*) para UART (*Universal Asynchronous Reception and Transmission*), simplificando a comunicação de outros módulos, com o microcontrolador, por meio da interface SPI.

O transporte dos dados pelo sistema, acontece em três situações principais: a primeira entre o *gateway* e os dispositivos, a segunda entre os componentes de hardware e a terceira entre o microcontrolador e o servidor local. Por fim, para o desenvolvimento da placa de circuito impresso e de todo o esquema de criação do diagrama elétrico no projeto, utilizou-se o software Proteus Design Suite.

3.2 Transceptor

O módulo transceptor (E32-915T30D), fabricado pela empresa Ebyte, emprega o chip da Semtech da série SX1276, e tem embutido em si a camada física do LoRa. Trata-se de um chip responsável pela comunicação sem fio, que proporciona comunicação em radiofrequência com proteção contra interferências e custo acessível (Semtech, 2019).

O módulo apresenta algumas especificações, entre as mais importantes está a capacidade de funcionar a uma frequência entre 900 e 931MHz, que por padrão se apresenta em 915MHz e consumo máximo de 1W. Além disso, é composto por sete pinos, entre eles existe dois que, de acordo com a combinação de bits, podem alterar o modo de operação (normal, *wake-up* ou *sleep*) (Ebyte, 2019). No modo de operação normal, tanto o canal sem fio quanto a comunicação serial estão disponíveis. O transmissor recebe os dados via serial e monta os pacotes a serem transmitidos ao receptor. Os pacotes possuem no máximo 58 bytes, para os dados que ultrapassarem esse valor, o mesmo, separa-se em vários pacotes. Entretanto, para os dados menores que 58 bytes, o módulo espera o tempo de 3 bytes e os trata como terminação de dados.

Pode-se alterar alguns parâmetros de configurações do módulo, sendo eles: HEAD, ADDH, ADDL, CHAN, SPED e OPTION. O parâmetro HEAD indica se as configurações realizadas anteriormente devem ser mantidas ou não quando o dispositivo for desligado ou reinicializado. Já os parâmetros ADDH e ADDL configuram o endereço do dispositivo. Para o parâmetro CHAN, modifica-se a frequência de comunicação. No parâmetro SPED, realiza-se três diferentes tipos de configuração: paridade da comunicação serial, taxa de comunicação UART (*Universal Asynchrounous Receiver/Transmitter*) e taxa de transmissão no ar. No parâmetro OPTION, define-se o tipo de transmissão (fixa ou

transparente), a utilização ou não de um resistor interno push-pull e pull-up, a potência de transmissão e o uso do *Forward Error Correction* (FEC), que garante a correção de erro de dados que sofreram determinado tipo de interferência.

Para este projeto, as configurações do tipo de transmissão, foram alterados para transmissão transparente e utilizou-se os parâmetros da seguinte forma: HEAD = 0xC0, CHAN = 0x0F caracterizando uma frequência de operação de 915MHz, bit de paridade UART 8N1 com taxa de 9600 bps, taxa de transmissão no ar de 300 bps, a potência de transmissão definida como 1W e o FEC ativado.

A transmissão transparente direciona os dados para todos os dispositivos que possuem o mesmo CHAN em suas configurações, ou seja, que operam em uma mesma frequência pré-configurada, independente do seu endereço.

3.3 Comunicação

A comunicação entre os dispositivos e servidor acontecem em diferentes etapas, tal como apresentado na Fig. 2.

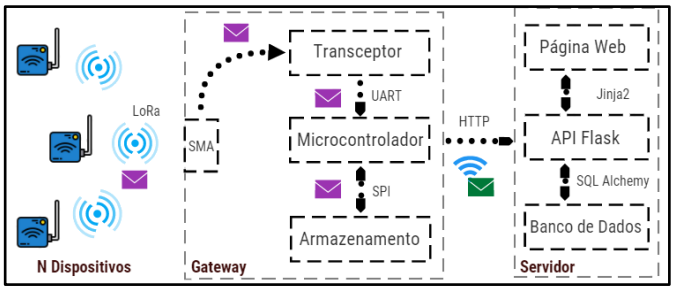

Fig. 2. Etapas da comunicação entre dispositivos LoRa e Servidor

Conforme Fig. 2, a comunicação entre os dispositivos e o gateway realiza-se por meio das técnicas de modulação de radiofrequência estabelecidas pela Semtech, utilizando protocolo próprio. Já entre os componentes de hardware acontecem em duas situações: a primeira entre o transceptor e o microcontrolador através da comunicação assíncrona, isto é, do protocolo UART e segunda entre o microcontrolador e o módulo de cartão micro SD por meio do protocolo SPI. Além disso, os dados são trafegados entre o microcontrolador e o servidor local através da rede sem fio (Wi-fi), utilizando solicitações HTTP via REST API. Os envelopes roxo e verde, presentes Fig. 2, representam as mensagens demonstradas nas seções *3.4* e *3.5*, respectivamente.

3.4 Protocolo

Dentro do formato do pacote da mensagem física estabelecido pela modulação LoRa está o *payload*, que carrega os dados de interesse. O *payload*, pode ocupar no máximo 58 bytes em uma transmissão sem que o pacote seja dividido. Nesse caso, determinou-se um campo que representa o MAC (*Media Access Control*) do dispositivo de origem, um campo que representa um instante único em que a mensagem é gerada (timestamp) e outro com os dados restantes, chamado de mensagem. A Fig. 3 ilustra o formato do pacote de mensagens.

Os campos são separados por um ponto de interrogação (?) e finalizado com o caractere especial (#), ocupando 1 byte cada. Os primeiros 17 bytes indicam o MAC e possuem o formato AA:AA:AA:AA:AA:AA, onde ‘A’ é um número ou letra. Os outros 10 bytes, após a separação, são dedicados para o horário e data de envio da mensagem. Por último, os 28 bytes restantes servem para carregar os dados que se deseja manipular.

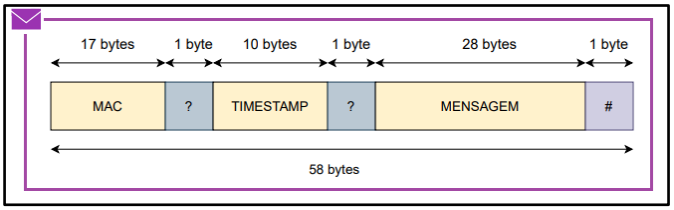

Fig. 3. Formato do pacote de mensagens.

3.5 Firmware

O firmware, executado pelo microcontrolador, responsabiliza-se pela inicialização do transceptor LoRa e do módulo micro SD, gerenciando os dados de recebimento e envio. Para isso, o firmware determina se os dados devem ser armazenados localmente em arquivos de texto (.txt) no cartão SD ou transmitidos, conforme o sucesso de envio. Caso o cartão SD fique cheio, as mensagens mais antigas passam a ser eliminadas do sistema.

Quando a mensagem chega ao *gateway* enviada pelo dispositivo, o seu MAC é adicionado junto a mensagem e enviado ao servidor em formato JSON (*JavaScript Object Notation*), conforme mostra o exemplo da Fig. 4.

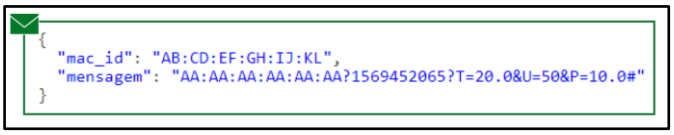

Fig. 4. Formato da mensagem em JSON.

Todo o desenvolvimento relacionado a comunicação de microcontrolador com outro dispositivo foi feito na linguagem C/C++ por meio do ambiente de desenvolvimento do Visual Studio Code (VS Code) utilizando a extensão PlatformIO, com o framework do Arduino e o sistema operacional de tempo real FreeRTOS. Além disso, esse conjunto traz vantagens como confiabilidade, portabilidade e segurança.

3.6 Software

Com o propósito de criar um servidor local com a capacidade de manipular simultaneamente diferentes solicitações, desenvolveu-se uma REST API. A implementação da REST API foi feita utilizando a linguagem Python com o VS Code, fazendo uso do micro framework Flask em conjunto com o Jinja e SQLite, conforme mostrado no diagrama da Fig. 5.

Para isso, habilitou-se o servidor WSGI (*Web Server Gateway Interface*) incluído no Flask, garantindo assim as requisições paralelas. Dessa maneira, quando uma solicitação é realizada, essa é direcionada para um novo segmento. Os dados são armazenados ou consultados no banco de dados, utilizando o SQLAlchemy, para mapeamento do objeto-relacional SQL e suas sessões. Os dados adquiridos, são enviados para o Jinja2 e renderizados com base no modelo HTML. A saída, é uma página HTML, gerada dinamicamente e mostrada no navegador conforme a URL da API acessada pelo usuário.

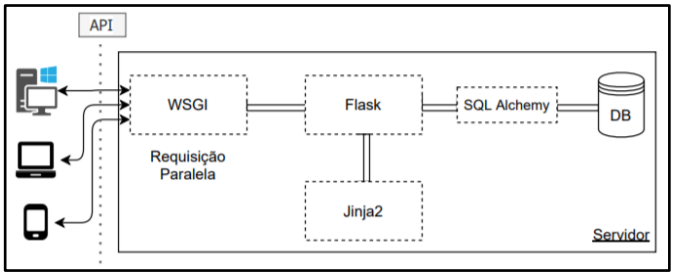

Fig. 5. Diagrama de software.

A preferência, pelo micro framework Flask, ocorre pela sua estrutura inicial bastante simples, que cresce conforme a necessidade do desenvolvimento, mas que ainda assim continua sendo poderoso e ideal para projetos pequenos que buscam simplicidade e rapidez.

3.7 Banco de Dados

Para desenvolvimento do banco de dados da aplicação, em SQLite, elaborou-se seis tabelas que se relacionam entre si por meio da chave estrangeira (*Foreign Key*), conforme apresentado na Fig. 6.

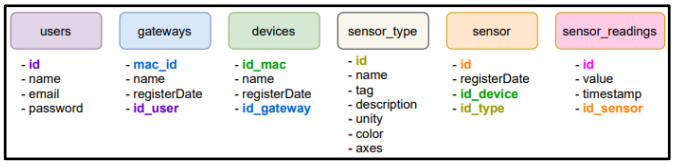

Fig. 6. Tabelas do banco de dados.

A tabela de usuários (*users*), contém todos os usuários cadastrados no sistema e garante o acesso restrito aos dados da aplicação. Além disso, tem-se a tabela de *gateways* e dispositivos (*devices*), que compõe a lista de *gateways* e dispositivos, respectivamente, espalhados pela fazenda, onde cada dispositivo está associado a um determinado gateway. Ao mesmo tempo, mantém-se a tabela de sensores (*sensor*), que está relacionado ao dispositivo. Os valores enviados por cada sensor, de um certo dispositivo, são armazenados na tabela de leitura de sensores (*sensor_readings*). Por fim, a tabela tipos de sensores (*sensor_type*), preenchida com valores fixos de diversos tipos de sensores na inicialização do sistema, permite que o sensor seja associado a um grupo específico.

3.8 Mecânica

Para a construção do projeto se fez necessário uma estrutura que seja apresentável ao usuário final e que possua tamanho razoável, de forma que pudesse ser fixado facilmente no ponto mais alto da fazenda. Além disso, uma demanda do projeto era que fosse de baixo custo, assim como o restante dos componentes de hardware. Dessa forma, desenvolveu-se o compartimento (*case*) utilizando o software de criação de modelos 3D SketchUp para ser impresso em uma impressora 3D, utilizando filamento PLA (Poliácido Láctico), conforme apresentado na Fig. 7.

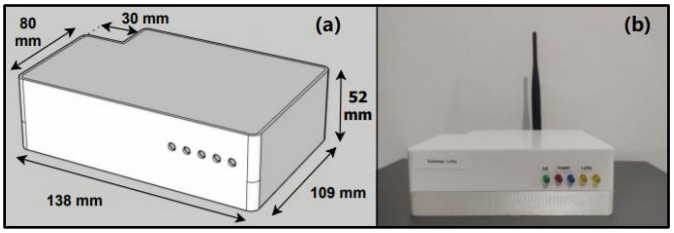

Fig. 7. *Gateway* LoRa: (a) Modelo 3D; (b) Resultado do *Case* em sua vista frontal.

4. EXPERIMENTOS

Esta seção apresenta os procedimentos e resultados de experimentos que demonstram a taxa de sucesso da transmissão de pacotes. Foram escolhidos sete locais diferentes a fim de diversificar o resultado das amostras onde são coletados os dados. Foi considerado o número de 100 mensagens enviadas com uma taxa de transferência de dados de 0,5 Hz, ou seja, um pacote a cada dois segundos.

O primeiro local escolhido foi o parque Barigui, localizado em Curitiba, Paraná. A extensão escolhida possui aproximadamente 1,1 Km com suas coordenadas localizadas em (-25,40750°; -49,41867°) e (-25,39629°; -49,42281°) chamado de experimento 1. A segunda região escolhida é localizada em Campo Magro, munícipio à cerca de 12 Km da cidade de Curitiba. Os pontos de coordenadas (-25,43084°; -49,30850°) e (-25,42047°; -49,30505°) possuem aproximadamente 1,4 Km em um terreno ondulado com baixa densidade populacional, descrito como experimento 2.

Os próximos cinco experimentos foram realizados ao longo da rodovia BR-376, localizada próximo à Curitiba, onde a distância máxima alcançada foi de 2,0 Km em uma rodovia bem movimentada e um terreno plano conforme Fig. 8.

Fig. 8. Mapa do experimento com maior distância, 2 Km.

Utilizou-se o trecho com distância de 2,0 Km para os experimentos 3 e 4, realizados em dias e horários diferentes,

com as coordenadas (-25,43028°; -49,3445°) e (-25,43143°; -49,32454°). Os experimentos 5 e 6 possuem uma extensão próxima a 1,5 Km em direções opostas com as coordenadas (-25,43028°; -49,3445°), (-25,43054°; -49,32965°) e (-25,43028°; -49,3445°), (-25,431136°; -49,359783°). Por fim, o experimento 7 possui uma distância de 1,9 Km, em direção oposta à coordenada do experimento 3 com as coordenadas (-25,43028°; -49,3445°), (-25,43171°; -49,36232°).

5. RESULTADOS

Neste estudo, uma demonstração dos resultados colhidos é evidenciada na Tabela 1, em função da distância e da taxa de extração de dados, que representa a divisão do número de mensagens enviadas com o número de mensagens recebidas.

Tabela 1. Taxa de extração de dados

Nº	Distância	Mensagens enviadas	Mensagens Recebidas	Taxa de extração
1	1,1 Km	450	445	98,89%
2	1,4 Km	57	47	82,46%
3	2,0 Km	100	94	94,00%
4	2,0 Km	100	83	83,00%
5	1,5 Km	100	100	100,00%
6	1,6 Km	100	99	99,00%
7	1,9 Km	100	98	98,00%

O experimento número 1 destacou-se devido sua altura relativa entre os dispositivos de aproximadamente 15 metros e uma observação direta entre eles. No experimento 2 observou-se uma grande dificuldade da propagação das ondas de rádio através de terrenos ondulados dos quais impediam uma visualização direta do dispositivo a ser comunicado, causando assim, a menor taxa de extração calculada dentre os experimentos. Os demais ambientes de teste possuem uma altura relativa próxima a zero, onde foi possível uma observação direta entre o dispositivo e o *gateway*.

6. CONCLUSÕES

Este artigo apresenta a construção de um *gateway* LoRa e servidor local para promover o acompanhamento das atividades concebidas no agronegócio, empregando a tecnologia de radiofrequência LoRa. O sistema apresenta melhorias para automatização que, quando aplicadas, proporcionam oportunidades de monitoramento das condições do campo de forma mais eficiente.

O protótipo foi projetado para receber mensagens via LoRa e transmiti-las ao servidor em ambientes abertos e descampados. O número de mensagens enviadas e recebidas durantes os testes são aceitáveis em uma distância próxima de 2 Km, no qual a perda de alguns pacotes não afetará severamente a aplicação. Uma limitação notada está em não ter um bom alcance quando o terreno é ondulado, de média densidade ou com dispositivos eletroeletrônicos causando interferência.

Em trabalhos futuros, pretende-se criar a topologia *wake-up*, ou seja, uma comunicação bidirecional entre o *gateway* e o dispositivo, onde o *gateway* fica responsável por requisitar a mensagem ao dispositivo. Bem como, realizar a otimização de parâmetros configuráveis de comunicação, por meio de algoritmos de aprendizado de máquina, dentre diferentes terrenos e condições ambientais, visando otimizar a taxa de extração de dados.

REFERÊNCIAS

Cepea (2021). PIB do Agronegócio. URL https://www.cepea.esalq.usp.br/upload/kceditor/files/Cepea_CNA_relatorio_2020.pdf.

Dias, E. (2016). Desmistificando REST com Java. URL https://s3.amazonaws.com/algaworks-assets/ebooks/algaworks-livreto-desmistificando-rest-com-java-1a-edicao.pdf.

Ebyte (2019). E32-915T30D User Manual. URL https://www.ebyte.com/en/downpdf.aspx?id=174.

Heble, S., Kumar, A., Prasad, K.V.V.D., Samirana, S., Rajalakshmi, P., and Dessai, U. B. (2018). A low power IoT network for smart agriculture. In *2018 4th World Forum on Internet of Things (WF-IoT)*, 609–614. IEEE.

Ikhsan, M. G., Saputro, M.Y.A., Arji, D. A. Harwahyu, R., and Sari, R. F. (2018). Mobile LoRa gateway for smart livestock monitoring system. In 2018 International Conference on Internet of Things and Intelligence System (IOTAIS), 46–51. IEEE.

Ji, M., Yoon, J., Choo, J., Jang, M., and Smith, A. (2019). LoRa-based visual monitoring scheme for agriculture IoT in 2019 Sensors Applications Symposium (SAS), 1–6. IEEE.

Ma, Y.W., and Chen, J.-L. (2018). Toward intelligent agriculture service platform with lora-based wireless sensor network. In 2018 International Conference on Applied System Invention (ICASI), 204–207. IEEE.

Oliveira, K.F.S. (2019). Conhecendo o Jinja2: um mecanismo para templates no Flask. URL https://imasters.com.br/desenvolvimento/conhecendo-o-jinja2-um-mecanismo-para-templates-no-flask.

Rodrigues, L.C.R. (2009). Arquitetura REST. URL http://monografias.ice.ufjf.br/tcc-web/exibePdf?id=17.

Santos, W. R. S. RESTful Web *Services* e a API JAX-RS. (2015). URL http://www.ricardoluis.com/wp-content/uploads/2015/08/Artigo-WebServices-em-REST.pdf.

Semtech (2019). LoRa Transceivers. URL https://www.semtech.com/products/wireless-rf/lora-transceivers/sx1276.

Souza, G.S.; Gomes, E.G.; Alves, E. (2020). Função de produção com base nos microdados do Censo Agropecuário de 2017. Revista de Política Agrícola, ano 29, 65-82. URL https://seer.sede.embrapa.br/index.php/RPA/article/view/1654.

The pallets project (2019). Flask. URL https://palletsprojects.com/p/flask/.