

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO DE JOINVILLE
CURSO DE ENGENHARIA MECATRÔNICA

CAUÊ VAL ARRUDA

ANÁLISE DA IMPLANTAÇÃO DE UMA REDE LORAWAN PARA SISTEMAS
EMBARCADOS

Joinville

2019

CAUÊ VAL ARRUDA

**ANÁLISE DA IMPLANTAÇÃO DE UMA REDE LORAWAN PARA SISTEMAS
EMBARCADOS**

Trabalho Conclusão do Curso apresentado como requisito parcial para obtenção do título de bacharel em Engenharia Mecatrônica no curso de Engenharia Mecatrônica, da Universidade Federal de Santa Catarina, Centro Tecnológico de Joinville.

Orientador: Dr. Anderson W. Spengler

Joinville

2019

AGRADECIMENTOS

Primeiramente gostaria de agradecer a minha família por toda a minha educação de casa, pelos ensinamentos sobre a vida, por todo o suporte no decorrer dos anos e principalmente por todo o amor e carinho oferecidos.

A minha namorada pelo apoio e incentivo em manter-me firme em momentos difíceis.

Aos meus amigos pelo companheirismo, troca de conhecimentos e pelos momentos juntos vividos, o dia a dia na universidade às vezes pode ser pesado, sem vocês a universidade não teria graça.

Aos professores do curso, em especial aos do laboratório LISHA, agradeço pelo esforço em repassar conhecimento, pela paciência quando exigida e também pelos momentos de descontração vivenciados.

“You’re tense, I’m calm. You apply excessive force; I control that force through fluid motion. That means relaxing the whole body so it can react instantly without resistance—no, without thought. Do you see now? It means becoming like clear water.”

— Spike Spiegel

RESUMO

Este trabalho aborda o estudo e a implantação de uma rede LoRaWAN, além de testes realizados para verificar a funcionalidade dela. Com o avanço da tecnologia, a presença de dispositivos inteligentes conectados junto da coleta de dados trazem qualidade de vida, produtividade e podem ajudar a prever condições de ambiente, e a cada dia que passa a Internet das Coisas (IoT) evolui junto, trazendo novas tecnologias além de um novo mundo conectado. Uma dessas tecnologias é LPWAN (Low Power Wide Area Network) capaz de conectar dispositivos a distâncias na ordem de quilômetros com um baixo consumo energético. Dentro das LPWAN, a tecnologia de rádio LoRa (Long Range) casada com o protocolo com modelo de negócio aberto desenvolvido para ela, a LoRaWAN, é uma das opções de LPWAN que mais chamam a atenção devido ao baixo custo, baixo consumo energético e longo alcance. Este trabalho propôs a implantação de uma rede LoRaWAN por completo, trazendo toda a arquitetura de rede e sua configuração com objetivo de estudo de seu funcionamento e análise de sua viabilidade. A configuração feita de forma simplista, mostrou que após a compreensão do funcionamento da rede, sua instalação e configuração não são tão complexas, além de trazer bons resultados de conectividade dentro de um condomínio industrial, chegando a distâncias de mais de 700 metros de cobertura de sinal.

Palavras-chave: Internet das Coisas. IoT. LPWAN. LoRa. LoRaWAN. Redes de sensores sem fio.

ABSTRACT

This work addresses the study and deployment of a LoRaWAN network, as well as tests performed to verify its functionality. Along the advancement of technology, the presence of smart devices attach and the use of data collection offers quality of life, and can help predict a view environmental conditions, and with each passing day the Internet of Things (IoT) evolved with new technologies beyond a new connected world. One of this technology is LPWAN (Low Power Wide Area Network) able to connecting devices at miles-ordering distances with low power consumption. Within LPWAN, the LoRa (Long Range) radio technology, coupled with the open business model protocol developed for it, the LoRaWAN, is one of the LPWAN options that draws more attention, with low cost, low power consumption and long range. This paper proposes the implementation of a complete LoRaWAN network, bringing all the network architecture and its configuration in order to study its operation and analyze its viability. A simplistic configuration has shown that after the network has been put into operation, its installation and configuration is not so complex, bringing good connectivity results within a business park, reaching a distance of over 700 meters of signal coverage.

Keywords: Internet of Things. IoT. LPWAN. LoRa. LoRaWAN. Wireless Sensor Network

LISTA DE FIGURAS

Figura 1 Comparação Distância vs Largura de Banda da LPWAN com outras tecnologias.	8
Figura 2 Orçamento de Link.....	12
Figura 3 Topologia estrela de estrelas.	14
Figura 4 Arquitetura de rede LoRaWAN	15
Figura 5 Estrutura do uplink da camada física	16
Figura 6 Estrutura do downlink da camada física	16
Figura 7 Janela de downlink Classe A	17
Figura 8 Classes de nós LoRa	18
Figura 9 Formato do DevAddr	19
Figura 10 Criptografia de pacotes LoRaWAN para impedir ataques de interceptação e repetição.	20
Figura 11 Arquitetura da Rede	22
Figura 12 Frente e verso do módulo RHF76-052.....	23
Figura 13 Placa com módulo LoRa e Antena.....	23
Figura 14 Arduino MEGA 2560.....	26
Figura 15 Módulo GPS NEO-6M.....	27
Figura 16 Conexões entre RHF0M301 e Raspberry Pi	29
Figura 17 Arquitetura do LoRa Server	32
Figura 18 Interface LoRa App Server	34
Figura 19 Interface de configuração do LoRa Gateway OS.....	35
Figura 20 Seleção do módulo concentrador	36
Figura 21 Seleção do padrão de canal	36
Figura 22 Seleção do bloco de canais.....	37
Figura 23 Configuração packet-forwarder	37
Figura 24 Configuração do gateway_ID	38
Figura 25 Tela inicial LoRa Gateway OS	39
Figura 26 Monitoramento do Status dos módulos do LoRa Server OS.	39
Figura 27 Monitoramento do Status dos módulos do LoRa Server OS.	40
Figura 28 Monitoramento do Status dos módulos do LoRa Server OS.	40
Figura 29 Gateways disponíveis na rede.	41
Figura 30 Adição de um gateway na rede	42

Figura 31 Criação do perfil do dispositivo final.....	43
Figura 32 Perfil de dispositivo.	44
Figura 33 Criação da Aplicação na rede.....	45
Figura 34 Configuração da Aplicação.	45
Figura 35 Criação do Dispositivo.....	46
Figura 36 Configuração de Ativação.....	47
Figura 37 Resposta do Arduino.....	49
Figura 38 Dados recebidos pelo servidor.	50
Figura 39 Locais de onde houve comunicação entre dispositivo final e gateway.....	52

LISTA DE QUADROS

Quadro 1 Símbolos de ações dos comandos AT	24
Quadro 2 Lista de comandos AT	25
Quadro 3 Ligações entre Arduino e modem LoRa.....	27
Quadro 4 Conexões entre RaspberryPi e Módulo RHF0M301	29

LISTA DE ABREVIATURAS E SIGLAS

IoT	Internet das Coisas
LoRa	Long-Range
LoRaWAN	LoRa for Wide Area Networks
LPWAN	Low Power Wide Area Networks
LISHA	Laboratório de Integração Software/Hardware
UFSC	Universidade Federal de Santa Catarina
SO	Sistema Operacional
RSSI	Indicador de intensidade do sinal recebido

SUMÁRIO

1 INTRODUÇÃO	7
1.1 OBJETIVOS	9
1.1.1 Objetivo Geral.....	9
1.1.2 Objetivos Específicos	9
1.2 ESTRUTURA DO TRABALHO	9
2 FUNDAMENTAÇÃO TEÓRICA.....	11
2.1. INTERNET DAS COISAS	11
2.2 LPWAN	11
2.3 LORA	13
2.4 LORAWAN.....	13
2.4.1 Classes de Dispositivos finais e tipos de mensagens	16
2.4.2 Ativação na rede e configurações	18
3 MATERIAIS E MÉTODOS.....	21
3.1 REQUISITOS DE PROJETO	21
3.2 ARQUITETURA DA REDE E SEUS COMPONENTES DE HARDWARE	21
3.2.1 Dispositivo final - Módulo LoRa RHF76-052	22
3.2.2 Dispositivo final - Arduino MEGA 2560	25
3.2.3 Dispositivo final - Módulo GPS NEO-6M	27
3.2.4 Gateway - Módulo RHF0M301	28
3.2.5 Gateway - Raspberry Pi 2B	28
3.3 ARQUITETURA DA REDE E SEUS COMPONENTES DE SOFTWARE - SISTEMA OPERACIONAL NO RASPBERRY PI E SOFTWARES UTILIZADOS	29
3.3.1 LoRa Gateway OS - Componentes do Sistema	30
3.3.2 LoRa Server	32
3.3.3 LoRa App Server	33
3.3.4 InfluxDB	34
3.4 INTEGRAÇÃO DO SISTEMA E CONFIGURAÇÕES	35
3.4.1 Configuração do LoRa Gateway OS.....	35
3.4.2 Acesso ao Gateway e sua adição na rede LoRaWAN.....	40
3.4.3 Criação de um Dispositivo Final na rede e de uma Aplicação	42
4 TESTES E RESULTADOS	48

4.1 SOFTWARE DE TESTE DO DISPOSITIVO FINAL E VALIDAÇÃO DO FUNCIONAMENTO DA REDE LORAWAN.....	48
4.1.1 Resultados do Teste	48
4.2 TESTE DE COBERTURA DE SINAL DENTRO DO PERINI BUSINESS PARK	50
4.2.1 Resultados do Teste	52
5 CONCLUSÃO.....	54
REFERÊNCIAS	56
APÊNDICE A	59
APÊNDICE B.....	62
ANEXO A.....	63

1 INTRODUÇÃO

A internet das coisas (IoT) é um paradigma tecnológico concebido como uma rede global de máquinas e dispositivos capazes de interagir entre si (LEE; LEE, 2015). Madakam, Ramaswamy, Tripathi (2015), complementa esse conceito com a ideia de que os dispositivos inteligentes possuem a capacidade de se auto-organizar, compartilhar informações, dados e recursos, podendo agir de maneiras diferentes dependendo da situação. Esse conceito ainda não é claro para todos os usuários, mas a tecnologia já está presente no cotidiano das empresas e pessoas (MADAKAN, RAMASWAMY e TRIPATHI, 2015). Muitas dessas aplicações trazem comodidade e economia em grande escala, como exemplo de uso, pode-se citar casas inteligentes, rastreamento de componentes em estoque, coleta de dados de umidade em diferentes pontos de uma plantação e outras aplicações de sensoriamento e acessórios inteligentes (BORBA, 2018).

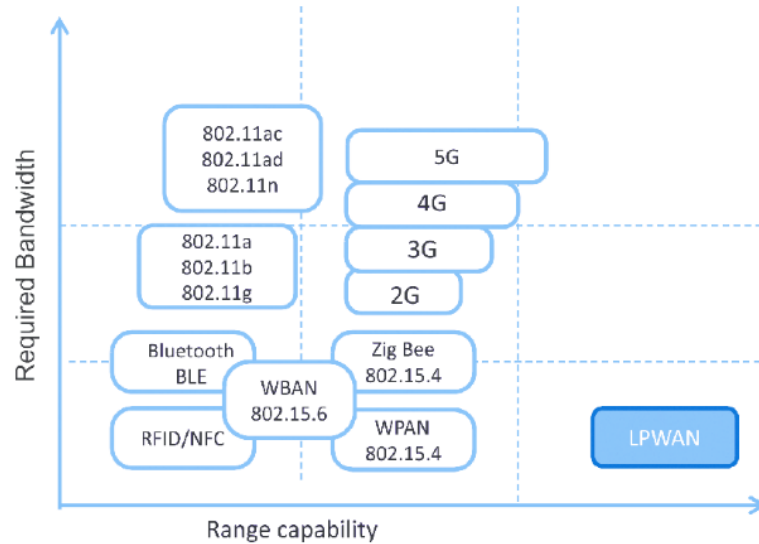
As aplicações de IoT podem ser divididas em indoor e outdoor. O grande diferencial entre elas é a distância de comunicação alcançada, podendo, por vezes, ter o mesmo objetivo e aplicar a mesma tecnologia de comunicação (ANGELIS et al., 2015). Exemplos de aplicações indoor são: controle automático de temperatura de ambientes e sistema de iluminação. Quanto às aplicações outdoor, é possível citar agricultura de precisão, sensoriamento de rios e outros locais de interesse. (TEIXEIRA; ALMEIDA, 2017)

Para aplicações outdoor de longo alcance, uma das alternativas de rede é a “Wide Area Network” (WAN), que possui tecnologias bastante difundidas no nosso dia a dia, como a rede de celular (e.g., 2G, 3G, e 4G), com infraestrutura instalada e grande área de cobertura (MEKKI et al., 2019). Ainda de acordo com Mekki et al. (2019) as soluções baseadas em comunicação de celular apesar de proverem boa área de cobertura, consomem muita energia do dispositivo. Devido à necessidade de economia de energia, foi impulsionado o surgimento de uma nova tecnologia de comunicação sem fio: “Low Power Wide Area Network” (LPWAN).

Da mesma forma que as redes de celular, as tecnologias LPWAN são caracterizadas por links de longo alcance (na ordem de quilômetros) e possuem topologias de rede em estrela, com os nós periféricos conectados diretamente a um concentrador que atua como gateway para o Protocolo da Internet (VANGELISTA; ZANELLA; ZORZI, 2015), se diferenciando da tecnologia GSM em quesitos como a busca do baixo consumo de energia, por isso as LPWAN possuem largura de banda limitada (MARQUES; BOCHIE, 2019). A

Figura 1, abaixo, apresenta uma comparação entre largura de banda e distância da LPWAN com as demais tecnologias.

Figura 1 Comparação Distância vs Largura de Banda da LPWAN com outras tecnologias.



Fonte: Haidine et al. (2016).

De acordo com Mekki apud Centenaro (2019), as LPWAN estão ganhando popularidade tanto no ramo industrial quanto entre pesquisadores devido ao baixo consumo, grande cobertura e baixo custo de comunicação, se comparada às outras tecnologias sem fio. Alcançando distâncias que variam de 10 a 40 km (dez a quarenta quilômetros) em áreas rurais e de 01 a 05 km (um a cinco quilômetros) em zonas urbanas. As duas tecnologias LPWAN mais difundidas são: LoRa for Wide Area Networks (LoRaWAN) e Sigfox. (SANTOS et al., 2107)

Santos et al. (2107) explica que o SigFox utiliza a tecnologia Ultra Narrow Band (UNB), projetada para lidar com pequenas taxas de transferências de dados. Por isso, promete um alcance de até 50 km em zonas rurais. Essa tecnologia trabalha com uma taxa de transmissão de dados que varia entre 10 bps e 1000 bps, operando na faixa de 900 MHz.

De acordo com Lora-Alliance (2015), o protocolo LoRaWAN foi projetado visando otimizar a vida útil das LPWANs. Além da capacidade, alcance e consumo de bateria, a tecnologia possui um modelo de negócio aberto e sua camada de rede trabalha em cima da camada física LoRa.

Segundo Fehri (2018) a tecnologia LoRaWAN vem chamando a atenção por ser uma tecnologia LPWAN sem fio de baixa potência e ampla cobertura de rede, considerada uma

das melhores alternativas de IoT devido ao baixo custo, baixo consumo energético e modelo de negócio aberto, além de fácil implementação estrutural.

Assim, esse estudo busca apresentar em termos gerais o protocolo LoRaWAN. Mostrando uma abordagem teórica dos termos mais comuns, e uma experimentação de uma forma de instalação e configuração de toda a arquitetura de rede e suas aplicações. O estudo considerará pontos positivos e negativos da implantação de LoRaWAN como proprietário da rede, versus a utilização de servidores externos.

A metodologia deste trabalho pode ser definida como descritiva quanto aos objetivos, uma vez que apresenta a tecnologia LoRaWAN com a intenção de esclarecer e familiarizar os leitores quanto ao tema. Esse estudo foi desenvolvido a partir de abordagem qualitativa de periódicos científicos com complementação de testes práticos aplicados à implantação da rede. A aplicação do sistema foi desenvolvida no Laboratório de Integração Software Hardware (LISHA) do Centro Tecnológico de Joinville (CTJ) da Universidade Federal de Santa Catarina (UFSC).

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Implantar uma rede LoRaWAN.

1.1.2 Objetivos Específicos

- Conhecer e explicar a tecnologia LoRa e o protocolo LoRaWAN;
- Analisar qualitativamente a dificuldade da implantação da rede LoRaWAN;
- Implantar e configurar as aplicações necessárias para a comunicação do protocolo LoRaWAN bem como integrar sistemas que possam ser úteis à rede;
- Testar e avaliar o sistema na prática.

1.2 ESTRUTURA DO TRABALHO

Esse trabalho está dividido em 6 capítulos. O primeiro introduz o tema e apresenta os objetivos desse estudo, além de indicar em termos gerais o que será apresentado nos capítulos

subsequentes. O segundo capítulo apresenta uma fundamentação teórica para o trabalho, indicando, a partir de periódicos científicos, os principais termos e conceitos para a melhor compreensão do assunto. O capítulo 2 divide-se em três subseções que explicam, respectivamente, IoT, LoRa e LoRaWAN.

No capítulo 3 podem ser encontradas com maiores detalhes a metodologia de estudo e a apresentação dos componentes do sistema considerado nesse trabalho. O capítulo 4 consiste da implantação prática do sistema, mostrando uma maneira simples de se implantar uma rede e colocar em operação. No capítulo 5 são feitos os testes a fim de avaliar a funcionalidade da rede e seu alcance, finalizando com o capítulo 6 onde é feita a análise da rede e as considerações finais do trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

2.1. INTERNET DAS COISAS

Internet das Coisas, do inglês Internet of Things (IoT) pode ser descrita como a maneira com que objetos físicos do nosso cotidiano são interconectados, onde muitos destes possuem incorporados uma inteligência ubíqua e são controlados pela internet. (COSTA; OLIVEIRA; MÓTA, 2018)

Espera-se que a Internet das Coisas seja a próxima revolução quando se trata do ecossistema móvel. É previsto um número em torno de 30 bilhões de dispositivos conectados até 2025, dos quais 7 bilhões, em torno de 23% do total, seriam módulos de Ampla-Cobertura Baixa-Potência (LPWAN) (NOKIA, 2015). Segundo Lee (2015), as aplicações IoT devem permitir interações de dispositivos entre si e entre humanos e, devem ser compostas de dispositivos confiáveis e robustos, garantindo a correta transmissão dos dados e permitindo o funcionamento adequado do sistema. O autor complementa dizendo que é importante que as aplicações de IoT sejam desenvolvidas com inteligência, de modo que os equipamentos possam monitorar o ambiente, identificar e resolver possíveis problemas com o mínimo de intervenção humana.

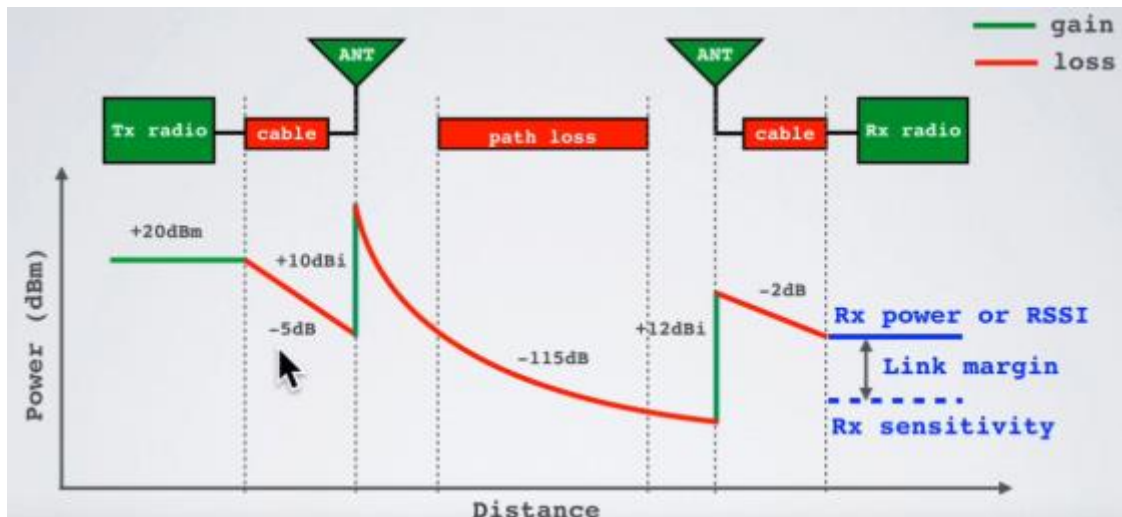
De acordo com Fehri (2018), desde o surgimento do conceito de IoT, foram propostas diversas tecnologias de comunicação entre aplicações, das quais destacam-se como tecnologia de redes de longo alcance, as Redes de Ampla-Cobertura e Baixa-Potência (LPWAN). As quais trazem como característica, junto ao baixo consumo energético, a baixa taxa de dados, além do fato de que foram construídas especificamente para IoT.

2.2 LPWAN

As tecnologias LPWAN se encaixam perfeitamente em situações nas quais os dispositivos precisam enviar pequenas quantidades de dados a longo prazo e, portanto, necessitam de uma bateria de longa duração (LINK LAB'S, 2016). Para que haja tal

comunicação de longo alcance, é preciso valor alto de orçamento de link¹ (Link Budget), conforme exemplificado na Figura 2.

Figura 2 Orçamento de Link



Fonte: B (2019)

O orçamento de link traz informações importantes principalmente quando se projeta sistemas de comunicação a rádio, pois permite projetar sistemas analisando a sensibilidade exigida do receptor a uma distância específica. Seu valor depende da distância, frequência e ganhos no sistema (BRUYNE et al., 2009).

Quando uma mensagem é transmitida, é necessário uma certa quantidade de energia para que seja detectada pelo receptor, uma parte da energia enviada é dissipada devido a interferências durante o caminho, como materiais e construções (LINK LAB'S, 2016). De acordo com Link Lab's (2016), as tecnologias LPWAN possuem uma característica interessante quanto a energia de recebimento de dados. A sensibilidade dos receptores opera em torno de -130 dBm, enquanto na maioria das tecnologias sem fio esse número fica entre -90 e -110 dBm. Essa diferença, quando analisada em potência de sinal, chega a detectar sinais 10.000 vezes mais fracos do que as tecnologias com -90 dBm, o que gera um grande ganho de distância de comunicação, já que o sinal pode perder bastante potência e ainda ser capturado. (LINK LAB'S, 2016)

As tecnologias LPWAN alcançam distâncias que variam de 10-40 km em áreas rurais e de 1-5 km em zonas urbanas. Essas distâncias são alcançadas devido às baixas taxas de transmissão, que resultam em uma potência de sinal maior por byte e consomem, por

¹ Orçamento de Link: é um termo de sistema de telecomunicações para o somatório de todos os ganhos e perdas do sinal de um transmissor para o receptor, através de um meio. (Fonte: <https://www.everythingrf.com/rf-calculators/link-budget-calculator>)

consequência, menos energia. Outro fator que ajuda no longo alcance é o fato de que frequências sub GHz (geralmente entre 415 MHz e 915 MHz) sofrem menos atenuação por obstáculos, se comparadas às tecnologias WAN que são transmitidas na ordem dos GHz (LIAGKOU; STYLIOS; SALMAS, 2019).

Como citado anteriormente as duas principais tecnologias LPWAN são a Sigfox e a LoRa. A Sigfox atua como uma operadora para IoT, com suporte a uma série de dispositivos, enquanto sua principal função é abstrair as dificuldades de conexão provendo uma API para as implementações de seus usuários (SANTOS et al., 2107).

Já a LoRa possui como característica destaque a utilização da modulação de rádio baseada em chirp spread spectrum (CSS), suas operações e arquitetura de rede são definidas pelo protocolo LoRaWAN.

2.3 LORA

A tecnologia LoRa, pertencente a camada física da tecnologia LPWAN, é uma modulação sem fio que oferece uma atraente combinação entre longo alcance, baixo consumo de energia e segurança na transmissão dos dados (SEMTECH, 2019).

Seu tipo de modulação se baseia no método de espalhamento espectral Chirp Spread Spectrum (CSS), na qual um símbolo é codificado em uma sequência longa de bits (chirp), reduzindo assim a relação sinal ruído e a taxa de interferência que o receptor precisa corrigir, sem alterar a largura de banda da frequência do sinal (SEMTECH, 2019).

A modulação CSS foi muito utilizada em aplicações militares devido a sua robustez à interferências e longo alcance de comunicação. Por conta do seu fator de propagação (Spread Factor) ser variável, é possível adaptá-lo em busca de uma taxa de transferência ideal para a aplicação. Sua alteração afeta a taxa de dados, que pode ser diminuída em troca de uma maior área de cobertura ou menor consumo de energia (CENTENARO et al., 2016).

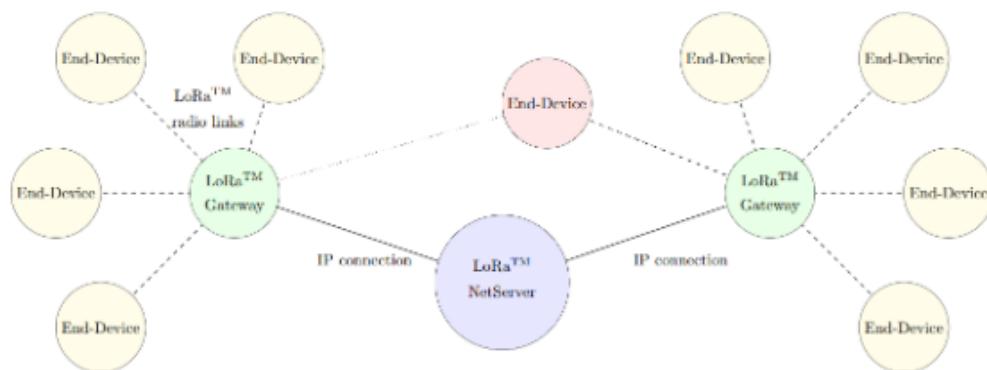
Diversos parâmetros são customizáveis na modulação LoRa, tais como Largura de Banda (BW), Fator de Propagação (SF) e Taxa de Codificação - Code Rate (CR). Tais parâmetros influenciam a taxa de bits efetiva da modulação, sua resistência ao ruído e sua facilidade de decodificação, tendo como principal parâmetro a largura de banda.

2.4 LORAWAN

Visando solucionar eventuais problemas de custo de gerenciamento de rede, organização, eficiência energética e cobertura e encontrar um protocolo de comunicação ótimo para a tecnologia LoRa, criou-se o protocolo LoRaWAN.

O protocolo LoRaWAN foi desenvolvido pela Cycleo e veio para padronizar o desenvolvimento de aplicações com LoRa. Esse protocolo fornece o mecanismo MAC (Media Access Control) que permite que diversos dispositivos periféricos se comuniquem com um único gateway (Augustin, 2016), tendo como característica de arquitetura da rede uma topologia estrela de estrelas demonstrada na Figura 3. Os periféricos não estão associados a um gateway específico, qualquer gateway registrado na mesma rede pode receber as mensagens de qualquer nó da rede, adicionando ao sistema maior garantia no recebimento de mensagens em áreas que coincidem a cobertura de mais de um gateway. (CENTENARO et al., 2016)

Figura 3 Topologia estrela de estrelas.



Fonte: CENTENARO et al. (2016).

Os concentradores, no momento em que recebem as mensagens dos nós via LoRa, retransmitem seus dados para o servidor de rede através da conexão com internet, utilizando o protocolo Internet Protocol (IP) e convertendo pacotes de Rádio Frequência (RF) em pacotes IP (Lora-Alliance, 2015).

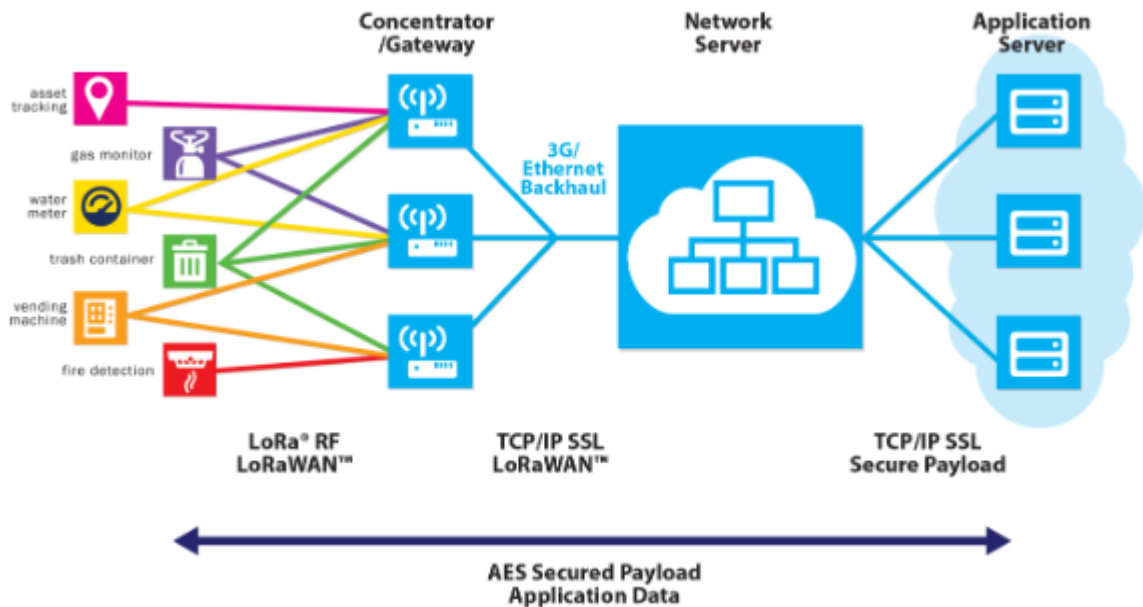
De acordo com Vangelista (2015), a arquitetura LoRa consiste em três principais componentes:

- Dispositivos Finais LoRa: Sensores e Atuadores conectados pela interface de rádio LoRa (Nós) a um ou mais Gateways.
- Gateways LoRa: Elemento central da rede, ponte entre os dispositivos finais e o servidor LoRa.

- LoRa Server: Servidor da rede LoRa, controla toda a rede (gerenciamento de recursos de rádio, controle de admissão, segurança, etc).

A arquitetura da rede pode ser vista na Figura 4 abaixo.

Figura 4 Arquitetura de rede LoRaWAN



Fonte: SEMTECH, 2018

A inteligência e complexidade são geridas pelo servidor de rede, que gerencia a rede, filtra pacotes redundantes, executa verificações de segurança, programa agendas de reconhecimento por meio do concentrador ideal e é capaz de executar taxa de dados adaptativa. Se um nó estiver em movimento não há a necessidade de troca de gateway, o que é um recurso crítico na usabilidade de dispositivos de rastreamento. (COMMITTEE, 2018)

Os nós de uma rede LoRaWAN são assíncronos e se comunicam quando estiverem prontos para enviar seus dados, sejam eles controlados por eventos ou programados, este tipo de protocolo também é conhecido como método Aloha. Em uma rede síncrona, como celular, os nós frequentemente precisam sincronizar com a rede e verificar se há mensagens, o que consome energia significativa do dispositivo e diminui a vida útil da bateria. (LoRa-Alliance, 2018)

Para tornar viável uma rede estrela de longo alcance, o gateway deve ter alta capacidade para receber mensagens de um número elevado de nós. Esta alta capacidade na rede LoRaWAN é alcançada utilizando taxa de dados adaptável e um transceptor multi-

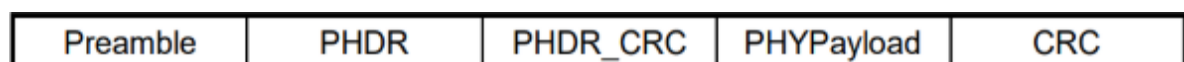
modem multicanal no gateway, assim, mensagens simultâneas em vários canais podem ser recebidas. Os fatores críticos que afetam a capacidade são o número de canais simultâneos, a taxa de dados (tempo no ar), o comprimento do dado e com que frequência os nós transmitem. (COMMITTEE, 2018)

Se um nó da rede possui um bom link e está posicionado próximo ao gateway, não é necessário configurá-lo na menor taxa de dados e ocupar o espectro disponível por um tempo maior do que o necessário. Ao aumentar a taxa de dados, o tempo no ar diminui, dando mais espaço para outros nós transmitirem, além disso essa taxa de dados adaptável pode otimizar a vida útil da bateria do nó. (COMMITTEE, 2018)

2.4.1 Classes de Dispositivos finais e tipos de mensagens

As terminologias das mensagens LoRa são distinguidas entre uplink e downlink. As mensagens Uplink são enviadas pelos dispositivos finais para o servidor de rede (Netserver) (COMMITTEE, 2018). O Netserver retransmite as mensagens por um ou mais gateways. Essas mensagens são estruturadas incluindo o cabeçalho da camada física do LoRa (PHDR), um cabeçalho CRC (PHDR_CRC) e um preâmbulo usado com propósito de sincronização. O tamanho do dado pode variar de 2 a 255 bytes e sua integridade é protegida por outro CRC. Os cabeçalhos e o dado são inseridos na mensagem pelo rádio transceiver (COMMITTEE, 2018). O formato do Uplink PHY pode ser visto na Figura 5.

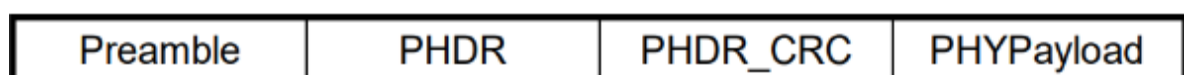
Figura 5 Estrutura do uplink da camada física



Fonte: COMMITTEE, 2018

Diferentemente do Uplink, o Downlink é enviado pelo Netserver para apenas um dispositivo final e entregue por apenas um gateway. O pacote de downlink inclui além do preâmbulo, o cabeçalho da camada física PHDR e um CRC do cabeçalho (PHDR_CRC), como no uplink. Seu formato é demonstrado na Figura 6.

Figura 6 Estrutura do downlink da camada física



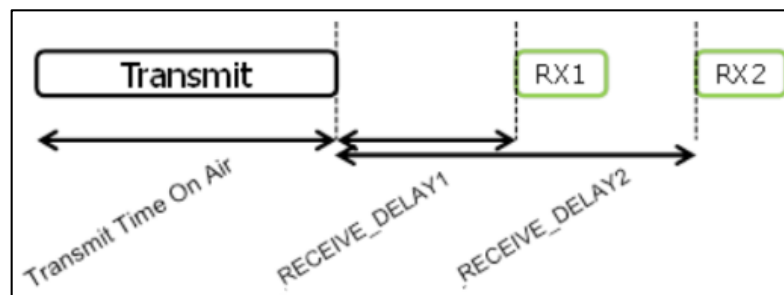
Fonte: COMMITTEE, 2018

Para buscar otimização e atender a diferentes perfis de aplicações, o LoRaWAN utiliza-se de três diferentes classes de dispositivos, as quais diferem-se principalmente na troca de latência da comunicação de downlink e duração da bateria. Normalmente o dispositivo terá suporte para mais de uma classe, que basicamente altera seu modo de funcionamento. A classe é escolhida durante a configuração da aplicação.

Dispositivos classe A: modo de funcionalidade padrão da rede LoRa, obrigatoriamente deve ser suportado por todos os dispositivos LoRa. É a classe com melhor desempenho energético, possui comunicação assíncrona e bidirecional (Protocolo Aloha) e é utilizada geralmente para aplicações de monitoramento, tendo a transmissão baseada na necessidade do dispositivo (LORA-ALLIANCE, 2019).

Nessa classe, após o envio da mensagem, o dispositivo abre duas janelas curtas para recebimento da mensagem de confirmação, utilizando-se como referência o final da transmissão. Seu baixo consumo energético é devido ao curto período de tempo da janela de recebimento, conforme a Figura 7 ilustra.

Figura 7 Janela de downlink Classe A



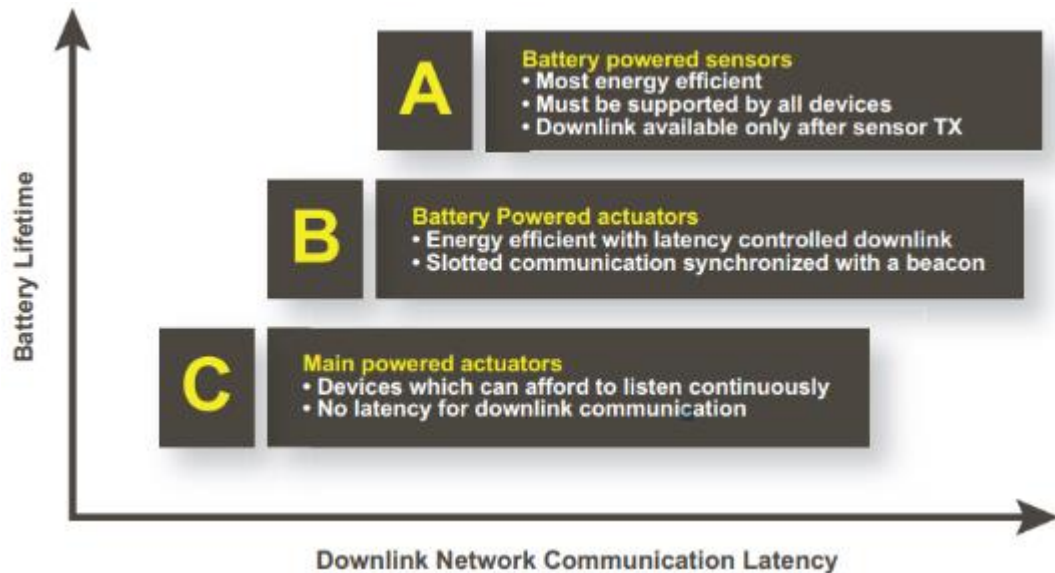
Fonte: COMMITTEE, 2018.

Dispositivos classe B: dispositivos finais são sincronizados com o servidor por meio de pacotes beacon através de gateways classe B, geralmente utilizados por dispositivos finais que precisam receber comandos de um controlador remoto ou prover dados requisitados pelo usuário. Na prática operam da mesma forma que dispositivos classe A, mas possuem janelas extras de downlink em horários programados, isso é possível pois ele recebe o sinal de sincronização do gateway, assim o servidor sabe quando o dispositivo final está escutando (MARQUES; BOCHIE, 2019).

Dispositivos Classe C: dispositivos finais que não possuem limitação de consumo energético, podem manter a janela de comunicação aberta, tendo como consequência o alto

consumo energético, pois a janela de recepção só é fechada durante a transmissão (VANGELISTA, 2015). A Figura 8, apresenta a comparação entre as classes e suas principais características.

Figura 8 Classes de nós LoRa



Fonte: LORA-ALLIANCE, 2015.

2.4.2 Ativação na rede e configurações

A fim de se conectar com a rede LoRaWAN, cada dispositivo deve ser configurado e ativado. Sua ativação pode ser feita de duas maneiras: Utilizando a Ativação pelo ar, OTAA (Over-The-Air Activation), feita quando um dispositivo é implantado ou redefinido; ou pela Activation By Personalization (ABP) “ativação personalizada”, ativação feita na etapa de configuração (COMMITTEE, 2018).

Após sua ativação, o dispositivo final passa a possuir um endereço de dispositivo (DevAddr), um identificador da aplicação (AppEUI), uma chave de sessão de rede (NwkSKey) e uma chave de sessão da aplicação (AppSKey). Melhor explicando:

- O DevAddr é formado por 32 bits que são utilizados para identificação do dispositivo final na rede. E segue o formato mostrado na Figura 9;

Figura 9 Formato do DevAddr

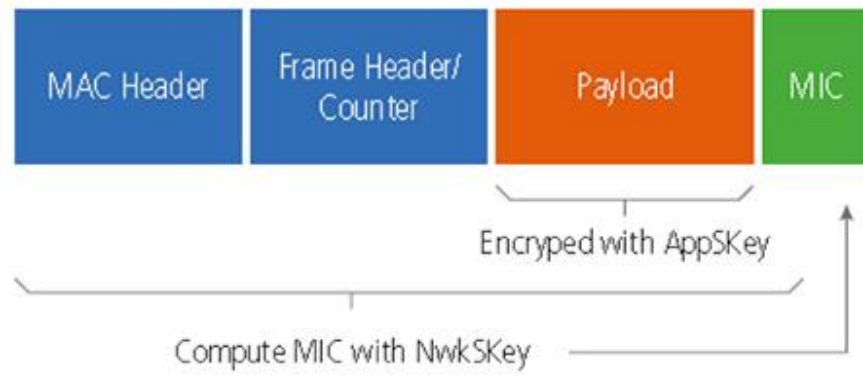
Bit#	[31..25]	[24..0]
DevAddr bits	NwkID	NwkAddr

Fonte: COMMITTEE, 2018.

Os 7 bits mais significativos são usados como identificador de rede (NwkID), separando em endereços possíveis redes no mesmo território além de solucionar possíveis problemas de roaming. O restante dos 25 bits menos significantes servem para indicar o endereço de rede do dispositivo final, o qual pode ser definido pelo administrador da rede (COMMITTEE, 2018).

- O AppEUI é um ID de aplicativo global no espaço de endereço IEEE EUI 64 (Extended Unique Identifier) que serve para identificar de maneira única o responsável pelo dispositivo. O AppEUI é armazenado no dispositivo final antes do processo de ativação ocorrer (COMMITTEE, 2018).
- O NwkSKey é a chave de sessão da rede específica de cada dispositivo final. Usada tanto pelo servidor de rede quanto pelo dispositivo final para calcular e verificar o código de integridade da mensagem e garantir a integridade dos dados. Além disso é usado para criptografar e descriptografar o campo de payload do MAC Layer (COMMITTEE, 2018).
- O AppSKey também é específico de cada dispositivo final, é usado pelo servidor de aplicação e pelo dispositivo final para criptografar e descriptografar o campo de payload da mensagem específica da aplicação. Ambas as chaves NwkSKey e AppSKey adicionam uma segurança ao pacote LoRaWAN a fim de impedir ataques de interceptação e repetição. Apesar disso, o pacote ainda não fica integralmente protegido, o servidor de rede apesar de ser considerado confiável, ainda pode alterar o conteúdo da mensagem em trânsito. Na Figura 10 pode ser vista a estrutura da mensagem e onde cada chave é utilizada (COMMITTEE, 2018).

Figura 10 Criptografia de pacotes LoRaWAN para impedir ataques de interceptação e repetição.



Fonte: EDITORS, 2017

3 MATERIAIS E MÉTODOS

Para o desenvolvimento do sistema será feita uma descrição das principais etapas do projeto, visando uma sequência organizacional. Na primeira etapa são definidos os requisitos do sistema, seu comportamento e o que ele deve fazer bem como o que deve ser avaliado na pesquisa. Na sequência, a seção 3.2 apresenta a etapa de descrição de arquitetura e componentes de hardware. Uma vez definidos os componentes de hardware necessários é possível passar para a fase de definição dos softwares utilizados, programação dos elementos do sistema e configuração dos elementos de rede, o que é mostrado na seção 3.3. Na seção 3.4 é mostrada a etapa de integração do sistema.

3.1 REQUISITOS DE PROJETO

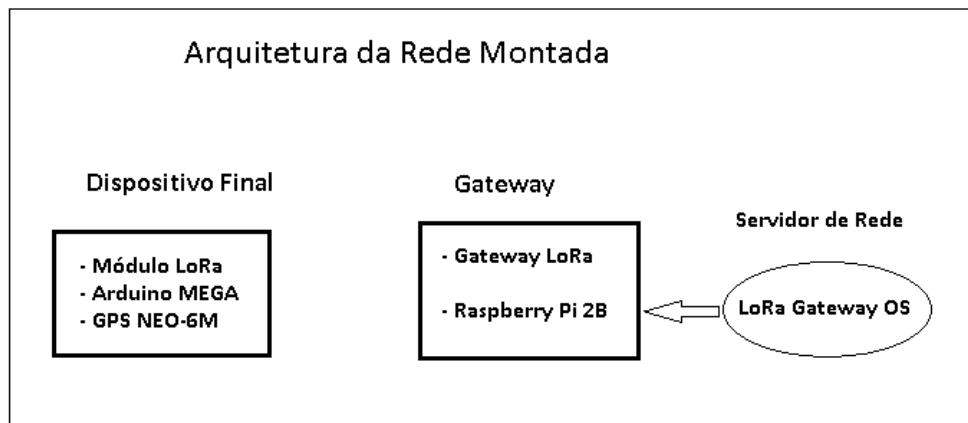
Com a finalidade de montar o sistema e colocá-lo em funcionamento, necessita-se fazer o levantamento de requisitos e o que será avaliado no projeto. Visando qualificar a utilização da plataforma LoRa em projetos de sistemas embarcados de cobertura local com objetivo de sensoriamento e sendo dono do próprio servidor de rede, serão analisados as seguintes variáveis qualitativas: dificuldade de implantação de todo o sistema, a distância de cobertura da rede LoRa implantada visando o mínimo investimento.

Para a implantação da rede LoRa, conforme apresentado na seção 2.4, são necessários 3 componentes principais: dispositivos finais LoRa, Gateway LoRa e servidor de rede. Para o dispositivo final é necessário um microcontrolador atuando sobre ele, além de adquirir os dados desejados para envio. Já para o Gateway é necessário um microprocessador rodando as aplicações de rede necessárias além de gerir o funcionamento do módulo de Gateway LoRa. O servidor de rede pode ser implantado em um computador.

3.2 ARQUITETURA DA REDE E SEUS COMPONENTES DE HARDWARE

Visando economia de custo no projeto e aproveitamento de componentes de hardware presentes no Laboratório em que o projeto foi feito (LISHA), foram escolhidos os seguintes componentes para compor a arquitetura de rede (ilustrada na Figura 11):

Figura 11 Arquitetura da Rede



Fonte: O Autor.

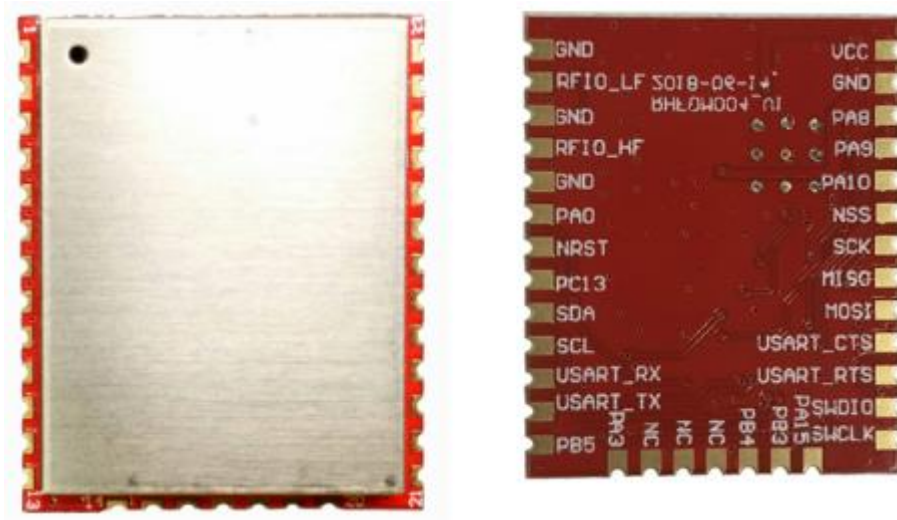
3.2.1 Dispositivo final - Módulo LoRa RHF76-052

Para a interface LoRa do dispositivo final foi escolhido o modem RHF76-052 da RisingHF. Esse módulo possui embarcado a pilha do protocolo LoRaWAN, facilitando assim sua configuração e utilização, sendo necessário para o usuário apenas o envio de comandos AT (RisingRF Users-Manual). Possui a opção de trabalho tanto com frequências baixas (434MHz / 470MHz) como com frequências altas (868MHz / 915 MHz). Devido a padrões de localidade e como o hardware do Gateway utilizado trabalha nesta faixa de frequência, foi utilizada a frequência de 915MHz.

Recomenda-se para o correto funcionamento do modem uma alimentação entre +1.8V e +3.6V, bem como manter sua temperatura entre -40 °C e +85 °C. Sua corrente de operação para o transceiver + MCU é de 45mA para a frequência de 915MHz. Nessa configuração a sensibilidade chega a -137dBm. O modem utilizado possui interfaces: USART, SPI, I2C, 8 GPIOs, RFIO_HF, RFIO_LF, USB, além de 2 entradas para ADC, e um pino para reset manual NRST. (RISINGHF, 2016).

Na Figura 122 é possível ver seu design, bem como todas as suas saídas.

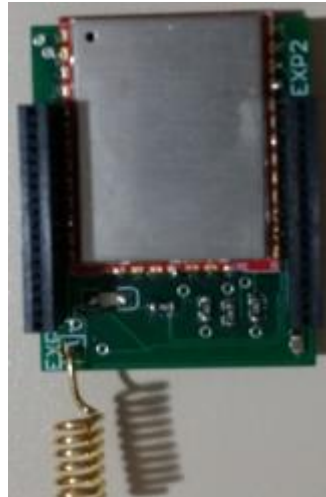
Figura 12 Frente e verso do módulo RHF76-052



Fonte: RF, 2018

Para facilitar conexões entre o arduino e o modem, foi utilizada uma placa de circuito impresso (PCB) para este módulo, desenvolvida dentro do LISHA. Na placa foi adicionada uma antena helicoidal que possui ganho de 4,5 dBi. Na Figura 13 pode ser vista a placa com o módulo e a antena.

Figura 13 Placa com módulo LoRa e Antena.



Fonte: O Autor.

3.2.1.1 Comandos AT do Módulo LoRa RHF76-052

Os comandos AT consistem em uma série de cadeias curtas de texto que podem ser combinadas para produzir comandos para diversas operações, utilizado por diversos dispositivos.

Esses comandos foram desenvolvidos em 1977 por Dennis Hayes e implementado no Hayes Smartmodem em 1981. Os comandos AT, também conhecidos como Comandos Hayes, foram desenvolvidos com o objetivo de criar uma interface para configuração e fornecimento de instruções para dispositivos (RIÇATO, 2018).

Segundo Risinghf (2016) as principais características a serem seguidas para envio dos comandos são:

- Os comandos são case insensitive, ou seja, não diferencia letras maiúsculas de minúsculas;
- Todos os comandos possuem resposta;
- Um comando nunca deve exceder 528 caracteres;
- Para que o comando AT seja válido deve terminar com ‘\n’ ou “\r\n”;
- Para comandos enviados pela UART, a configuração deve ser 9600 bits per second [bps], 8 bits, sem paridade, 1 stop bit.

Para a utilização dos comandos AT existem alguns símbolos importantes que são usados como identificadores de ações, seja para indicar que a partir do símbolo virá uma atribuição de valor, ou para separar parâmetros, entre outros símbolos. A Quadro 1 lista os símbolos utilizados pelos comandos AT do protocolo LoRaWAN e seus respectivos significados. (RISINGHF, 2016).

Quadro 1 Símbolos de ações dos comandos AT

=	Atribuição de valor para o comando
?	Pergunta para o dispositivo o valor do comando
:	Inicia uma lista de parâmetros de entrada
+	Prefixo do comando
,	Separador de parâmetros
Espaço	Character vazio, pode ser usado para formatar o comando.

Fonte: O Autor, baseado no Documento LoRaWAN Class A/C AT Command Specification

Após a breve apresentação sobre o funcionamento dos comandos AT, a Quadro 2 apresenta uma lista dos principais comandos presentes dentro da biblioteca de LoRaWAN utilizada no Arduino. O conhecimento dos comandos AT é importante para melhor compreensão de qual comando utilizar para a alteração de parâmetros e configuração do modem LoRa.

Quadro 2 Lista de comandos AT

Command	Description
AT	Test command
HELP	Print command list
FDEFAULT	Factory data reset
RESET	Software reset
DFU	Force bootloader to enter dfu mode
LOWPOWER	Enter sleep mode
VER	Version[Major.Minor.Patch]
MSG	LoRaWAN unconfirmed data
MSGHEX	LoRaWAN unconfirmed data in hex
CMSG	LoRaWAN confirmed data
CMSGHEX	LoRaWAN confirmed data in hex
PMSG	LoRaWAN proprietary
PMSGHEX	LoRaWAN proprietary in hex
CH	LoRaWAN channel frequency
DR	LoRaWAN datarate
ADR	LoRaWAN ADR control
REPT	Unconfirmed message repetition
RETRY	Confirmed message retry
POWER	LoRaWAN TX power
RXWIN2	LoRaWAN RX window2
RXWIN1	Customized RXWIN1 frequency
PORT	LoRaWAN communication port
MODE	LWABP, LWOTAA, TEST
ID	LoRaWAN DevAddr/DevEui/AppEui
KEY	Set NWKSKEY/APPSKEY/APPKEY
CLASS	Choose LoRaWAN modem class(A/B/C)
JOIN	LoRaWAN OTAA JOIN
LW	LoRaWAN misc configuration (CDR, ULDL, NET, DC, MC, THLD)
TEST	Send test serious command
UART	UART configure
REG	Control RF Chip
DELAY	RX window delay
VDD	Get VDD
RTC	RTC time get/set
EEPROM	Write/Read EEPROM

Fonte: RISINGHF, 2016

3.2.2 Dispositivo final - Arduino MEGA 2560

A plataforma escolhida para o desenvolvimento foi Arduino. Devido a vasta quantidade de informação disponível, facilidade de programação, comunidade bem desenvolvida e ampla aderência de utilização pelos entusiastas de sistemas embarcados.

Dentro da plataforma Arduino existem diversas opções de placas que utilizam microcontroladores diferentes, dependendo da aplicação desejada. Nesse estudo, considerando possível necessidade de aplicações futuras com a utilização de múltiplos módulos e sensores, foi definido o Arduino Mega 2560, mostrado na Figura 14. Essa placa utiliza o microcontrolador Atmega2560, que possui 54 pinos de entrada/saída digital (dos quais, 15 podem ser utilizados como saída PWM), 15 entradas analógicas, 4 UARTs, um cristal oscilador de 16MHz, conexão USB, entrada para fonte de alimentação, um cabeçalho ICSP e um botão de reset (ARDUINO, 2019).

Figura 14 Arduino MEGA 2560



Fonte: Arduino (2019)

3.2.2.1 Conexões Arduino - RHF76-052

Como será utilizada UART, optou-se pela porta serial 2 do Arduino, assim as ligações ficam conforme descreve a Quadro 3.

Quadro 3 Ligações entre Arduino e modem LoRa

Arduino Mega 2560 pin	RHF76-052 pin
RX2 - pin 17	USART1_TX
TX2 - pin 18	USART1_RX
3.3V	VCC
GND	GND

Fonte: O Autor

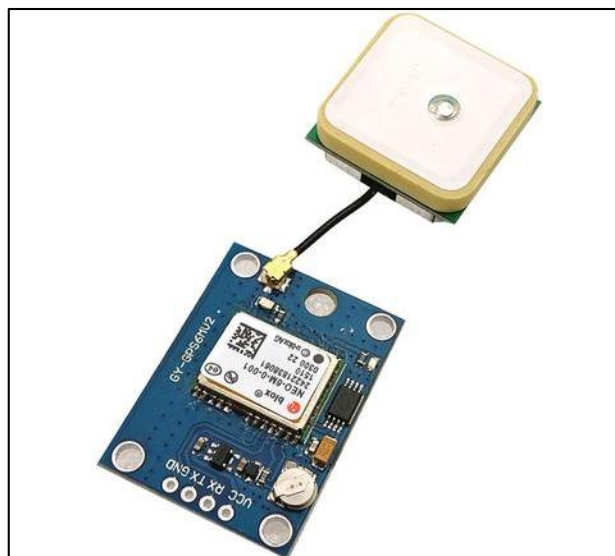
3.2.3 Dispositivo final - Módulo GPS NEO-6M

Para a realização do teste de distância, será utilizado o GPS NEO-6M, onde a distância será feita pelo cálculo da distância entre a coordenada do gateway e a coordenada do dispositivo final.

O módulo GPS NEO-6M (Figura 15) é um dispositivo simples de ser utilizado, além de barato, possui uma precisão de 5 metros e irá auxiliar no mapeamento da distância de envio do dispositivo final para o gateway.

O módulo possui tensão de operação de 3,3 volts e utiliza protocolo de comunicação Serial TTL - TxRx (U-BLOX, 2019).

Figura 15 Módulo GPS NEO-6M



Fonte: Twig (2019)

3.2.4 Gateway - Módulo RHF0M301

Para o módulo de gateway LoRa foi utilizado o módulo Gateway RHF0M301 da RisingRF, o qual possui o chip SX1301 da Semtech, projetado para oferecer recursos inovadores de gateway. Este módulo foi baseado na v1.0 LoRaWAN da Semtech e possui sensibilidade de até -140 dBm. Com uma alimentação de +5V, possui interface SPI e GPIO. Sua temperatura de operação deve permanecer entre -40 e +85 °C para o correto funcionamento (RISINGRF, 2019).

3.2.5 Gateway - Raspberry Pi 2B

Para fazer a interface do gateway, foi utilizado um componente mais robusto e com capacidade de execução de um Linux embarcado. Por ser uma plataforma bem conhecida, fácil acesso no mercado e por atender aos requisitos de componente, foi escolhido o Raspberry Pi.

O Raspberry Pi conta com processador ARM Cortex-A7, (processamento 900MHz quad-core), 1GB de memória RAM e possui processamento e memória suficientes para a execução de um Sistema Operacional (SO) baseado em Linux (FOUNDATION, 2019).

3.2.4.1 Conexões Raspberry Pi - RHF0M301

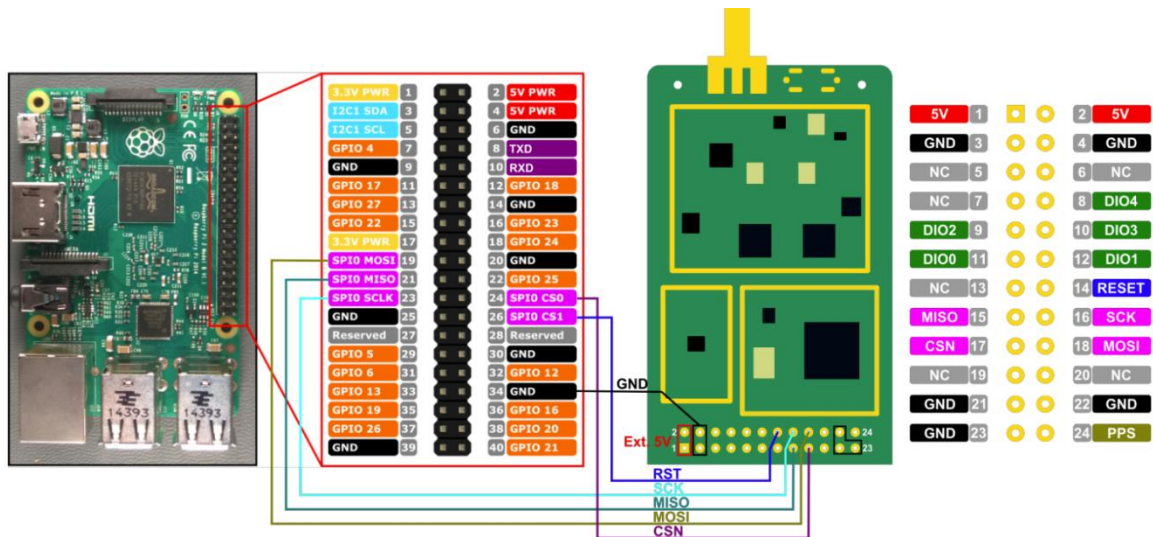
Dando sequência à instalação física do gateway, o Raspberry Pi foi conectado via interface SPI com o gateway LoRa, as conexões ficaram conforme descreve a Quadro 4 e podem ser melhor observadas na Figura 16.

Quadro 4 Conexões entre RaspberryPi e Módulo RHF0M301

RHF0M301 pin	Description	RPi physical pin
1,2	Supply 5V	2
3,4,21,22,23	GND	6
14	Reset	22
16	SPI CLK	23
15	SPI MISO	21
18	SPI MOSI	19
17	SPI NSS	24

Fonte: BABGEI, 2017

Figura 16 Conexões entre RHF0M301 e Raspberry Pi



Fonte: RisingHF (2018)

3.3 ARQUITETURA DA REDE E SEUS COMPONENTES DE SOFTWARE - SISTEMA OPERACIONAL NO RASPBERRY PI E SOFTWARES UTILIZADOS

Para a devida instalação e funcionamento do gateway, são necessários softwares responsáveis pela ligação entre os sistemas, além de um SO para execução dos softwares. Após a verificação das possibilidades, decidiu-se aplicar os componentes de rede desenvolvidos pelo LoRa Server Project, um projeto de código aberto que fornece componentes para a construção de redes LoRaWAN. Estes componentes formam uma solução pronta para uso, incluindo uma interface web fácil de usar e APIs gRPC e REST. Além disso, os componentes também podem ser personalizados ou integrados a LoRa Server em

infraestruturas existentes. Todos os componentes possuem a licença MIT, e podem ser usados para fins comerciais (CHIRPSTACK, 2019).

Definida a base de componentes necessária, foi escolhido um Sistema Operacional para suportar os programas baseado em Linux. Assim foi escolhido o Raspbian por ser considerado o Sistema Operacional padrão da Raspberry Foundation (FOUNDATION, 2019) e possui ampla utilização pela comunidade de usuários do Raspberry Pi.

O Raspbian é um sistema operacional baseado em Debian e otimizado para rodar em Raspberry Pi (RASPBIAN, 2019). Foi desenvolvido de maneira independente e não filiada a Raspberry Pi Foundation, dedicado a fãs da plataforma e a desenvolvedores e contam também com o propósito educacional.

Após a instalação do Raspbian, dos programas necessários provindos do LoRa Server Project e feitos alguns testes do sistema, o sistema ainda não estava bom. O Raspbian conta com alguns pacotes adicionais instalados junto ao sistema. Muitos desses pacotes são desnecessários na aplicação considerada, e algumas das aplicações instaladas para a rede LoRa só inicializavam de maneira manual, também foi notado alguns problemas para inicializar o packet-forwarder. Após o início dos testes no Raspbian, foi lançado pelo LoRa Server Project o LoRa Gateway OS, feito para rodar no Raspberry Pi e no LORIX One, tem objetivo de facilitar configurações de Gateways LoRa que utilizem o LoRa Server (CHIRPSTACK, 2019), optou-se assim por utilizá-lo e analisar sua praticidade de instalação e configuração dos componentes de rede restantes, descontinuando o uso do Raspbian neste trabalho.

O LoRa Gateway OS é um sistema operacional open-source, também baseado em Linux, com o diferencial de já vir preparado com os componentes do LoRa Server Project exclusivamente para funcionar junto em Raspberry Pi fazendo a interface de Gateway LoRa. Nesse sistema operacional estão inclusos o LoRa Server e o LoRa App Server, exigindo o mínimo de instalações e configurações.

Antes da utilização do LoRa Gateway OS, o LoRa Server e o LoRa App Server haviam sido instalado em um computador do LISHA, adicionando complexidade para a configuração de todo o sistema, o LoRa Gateway OS já vem com o LoRa Server e o LoRa App Server pré-configurados para a correta comunicação com o packet-forwarder.

3.3.1 LoRa Gateway OS - Componentes do Sistema

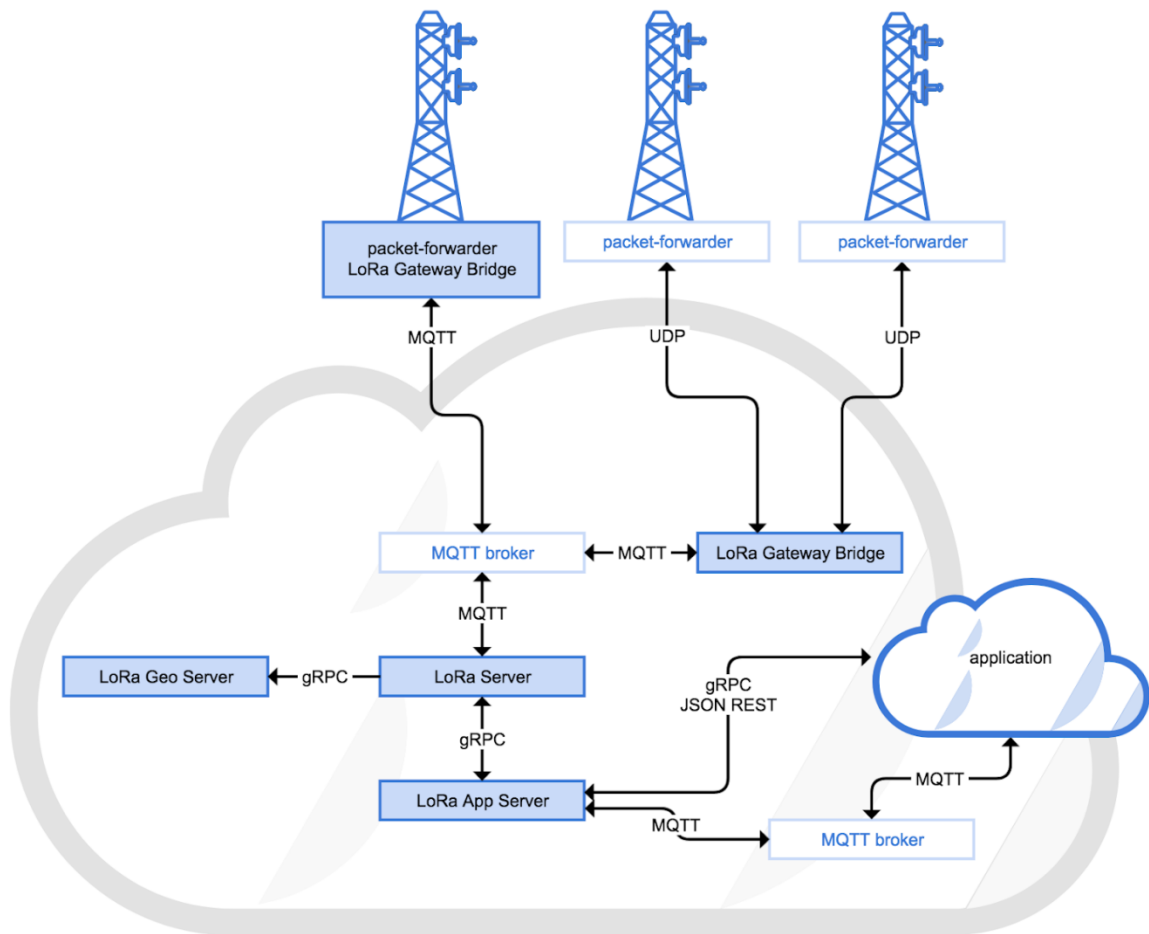
Com a utilização do LoRa Gateway OS, a complexidade da instalação e configuração do restante do sistema tornou-se simples, este SO já fornece o packet-forwarder da Semtech e o LoRa Gateway Bridge pré-instalados, incluindo um utilitário com interface por linguagem de comandos para auxiliar na configuração do gateway.

Para melhor compreensão do sistema são apresentados os principais componentes internos ao LoRa Gateway Bridge e programas responsáveis pelo protocolo de maneira resumida (CHIRPSTACK, 2019):

- *Packet-forwarder* é um programa executado no host do gateway LoRa, para encaminhamento dos pacotes RF recebidos pelo concentrador do gateway LoRa, através de um link IP/UDP para o servidor de rede;
- *LoRa Gateway Bridge* é um serviço que converte os pacotes do protocolo LoRa do packet-forwarder para os do protocolo comum do LoRa Server (JSON e Protobuf) fazendo parte fundamental do projeto LoRa Server;
- *MQTT (Message Queuing Telemetry Transport)* utilizado para fazer a interface entre o LoRa Gateway Bridge e o LoRa Server, é um protocolo de mensagens entre máquinas desenvolvido pela IBM no final dos anos 90, com o principal objetivo de ajudar na comunicação entre sensores e satélites, hoje em dia muito utilizado em aplicações IoT. Baseado no TCP/IP, suporta comunicações assíncronas (YUAN, 2017).

O LoRa Server e o LoRa App Server, também inclusos no LoRa Gateway OS, serão abordados nas seções seguintes. A Figura 17 apresenta uma disposição geral dos componentes do sistema operacional, bem como sua arquitetura.

Figura 17 Arquitetura do LoRa Server



Fonte: SERVER, 2019

3.3.2 LoRa Server

O LoRa Server é um Servidor de Rede de código aberto, também parte do LoRa Server Project. Sua responsabilidade é de manuseio dos quadros de uplink recebidos pelo gateway, e assim fazer o agendamento das transmissões de downlink. É responsabilidade do Server lidar com a autenticação dos dispositivos gerando as chaves de sessão.

O LoRa Server traz algumas funcionalidades interessantes quanto a otimização dos dispositivos, uma delas é a taxa de dados adaptável (ADR), que permite ao servidor controlar a taxa de dados e a potência TX do dispositivo, usando menos tempo de antena e consumindo menos energia ao transmitir a mesma quantidade de dados. Além do benefício da economia de energia do dispositivo há o benefício de otimização do espectro, valendo ressaltar que o ADR só é indicado para dispositivos estáticos e não deve ser usado para dispositivos que se movem (SERVER, 2019).

Outra funcionalidade interessante é a solicitação periódica do status do dispositivo, usando o DevStatusReq (comando mac especificado pelo protocolo LoRaWAN), em que um dispositivo ao responder a essa solicitação, envia o status da bateria (se disponível) e sua relação sinal/ruído de demodulação em dB para a última solicitação recebida com êxito (SERVER, 2019).

3.3.3 LoRa App Server

Também incluso no LoRa Gateway OS, o LoRa App Server é um servidor de aplicação que possui uma interface-web intuitiva e de fácil configuração, responsável pela parte do “inventário” dos dispositivos da infraestrutura do LoRaWAN. o LoRa App Server também lida com as solicitações de conexão e com a criptografia dos payloads das aplicações (SERVER, 2019).

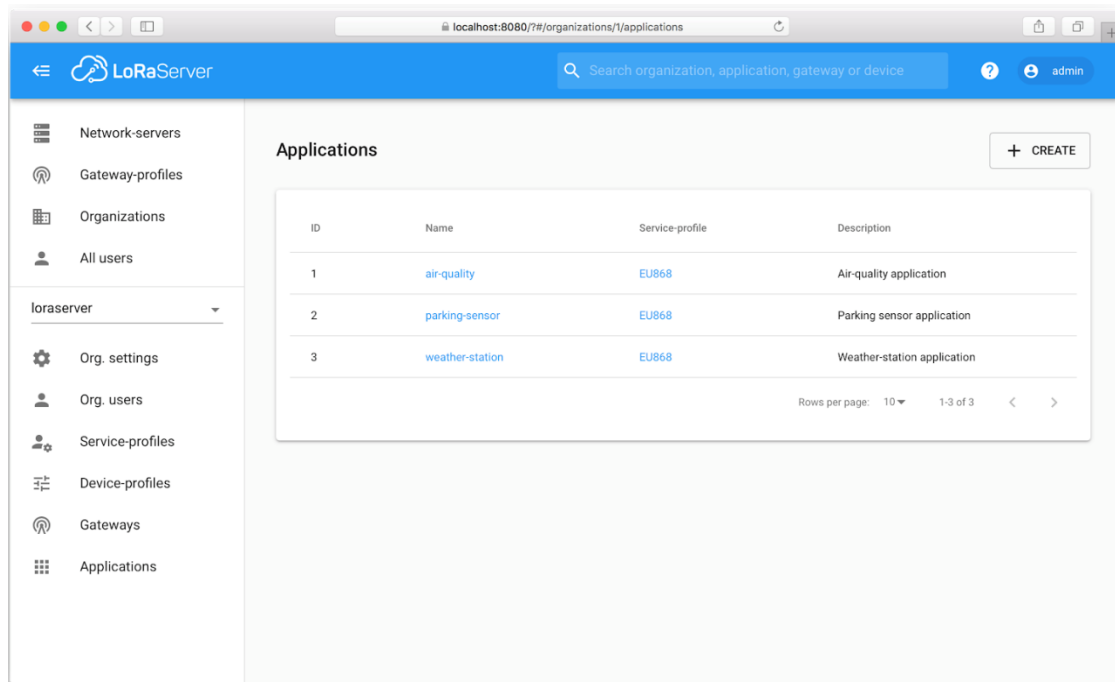
Além dos pontos citados acima, o LoRa App Server possui em sua interface a gerência de acesso ao App, trazendo login para permissão de visualização e configuração e permitindo conceder aos usuários permissões de administradores globais, administradores de organização ou permissão de “apenas visualização” do projeto. Isso possibilita executar o LoRa App Server em um ambiente multilocatário, onde cada organização ou equipe tem acesso apenas a seus próprios aplicativos e dispositivos (SERVER, 2019).

Relativo ao envio e recebimento de dados, o LoRa App Server traz diferentes possibilidades de integrações, tais como: MQTT, PostgreSQL, Serviço de Notificação Simples da AWS, Azure Service Bus e Plataforma do Google Cloud. Além destas integrações globais, também é possível configurar integrações pela aplicação. As integrações disponíveis na aplicação são: HTTP, InfluxDB e ThingsBoard (SERVER, 2019).

O LoRa App Server também conta com registro de eventos em tempo real e permite inspecionar os eventos que estão ocorrendo em tempo real, sem a necessidade de desenvolver qualquer integração. Porém esse “registro” não salva os dados recebidos, a não ser que o usuário salve manualmente ou a integração com banco de dados seja feita.

Na Figura 18 é possível observar a interface do LoRa App Server e a Seção 4 aborda seu acesso, configuração de permissões, configuração de dispositivos tais como Gateway e dispositivo final, além da interface de aplicação do dispositivo.

Figura 18 Interface LoRa App Server



Fonte: SERVER, 2019

3.3.4 InfluxDB

Como o LoRa App Server não armazena os dados de payload, optou-se por utilizar o InfluxDB para seu armazenamento. InfluxDB é um banco de dados de séries temporais (TSDB) de código aberto, desenvolvido pela InfluxData. O Influx DB foi desenvolvido na linguagem Go e otimizado para ser rápido, com alta capacidade de armazenamento e trazer fácil operação para monitoramento, como dados de sensores de IoT (INFLUXDATA, 2019).

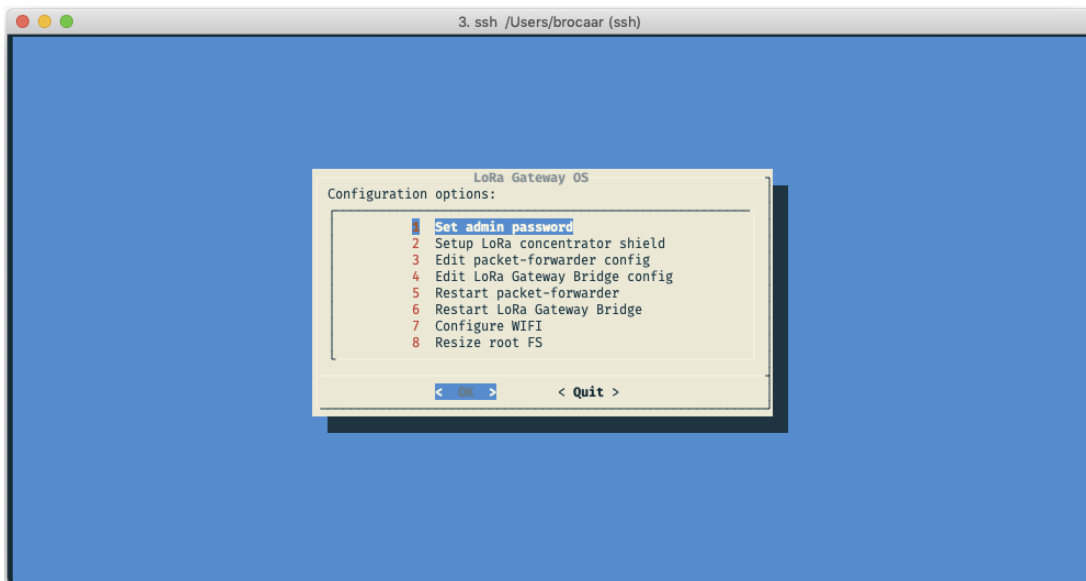
3.4 INTEGRAÇÃO DO SISTEMA E CONFIGURAÇÕES

3.4.1 Configuração do LoRa Gateway OS

Após a instalação do Lora Gateway OS no microSD do Raspberry Pi e sua inicialização é possível acessá-lo de duas maneiras, utilizando um monitor ou por acesso SSH, sendo a segunda, a forma mais utilizada. Na configuração padrão para o login inicial, ambos o usuário e senha são admin.

A primeira vez que o sistema é inicializado, é possível ver a tela de configuração do gateway, como mostrado na Figura 19.

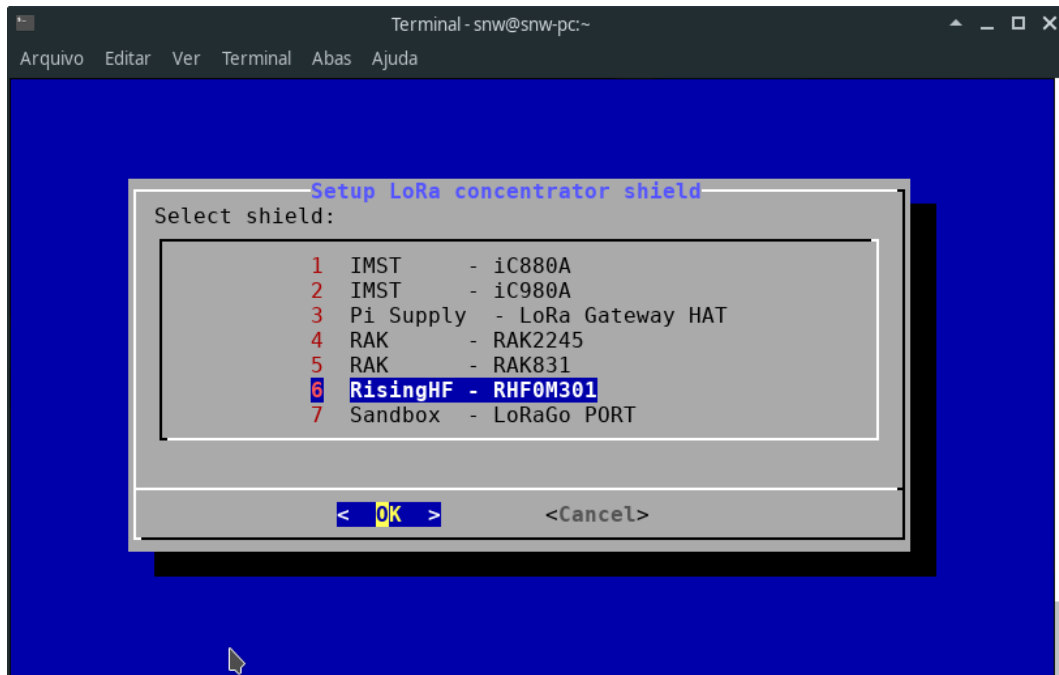
Figura 19 Interface de configuração do LoRa Gateway OS



Fonte: SERVER, 2019

Inicialmente é necessário configurar o concentrador a ser utilizado, na opção 2 - "Setup LoRa concentrator shield" estão as configurações do módulo. O primeiro item a ser selecionado é o concentrador (gateway), no caso, opção 6 RisingHF - RHF0M301, como demonstrado na Figura 20.

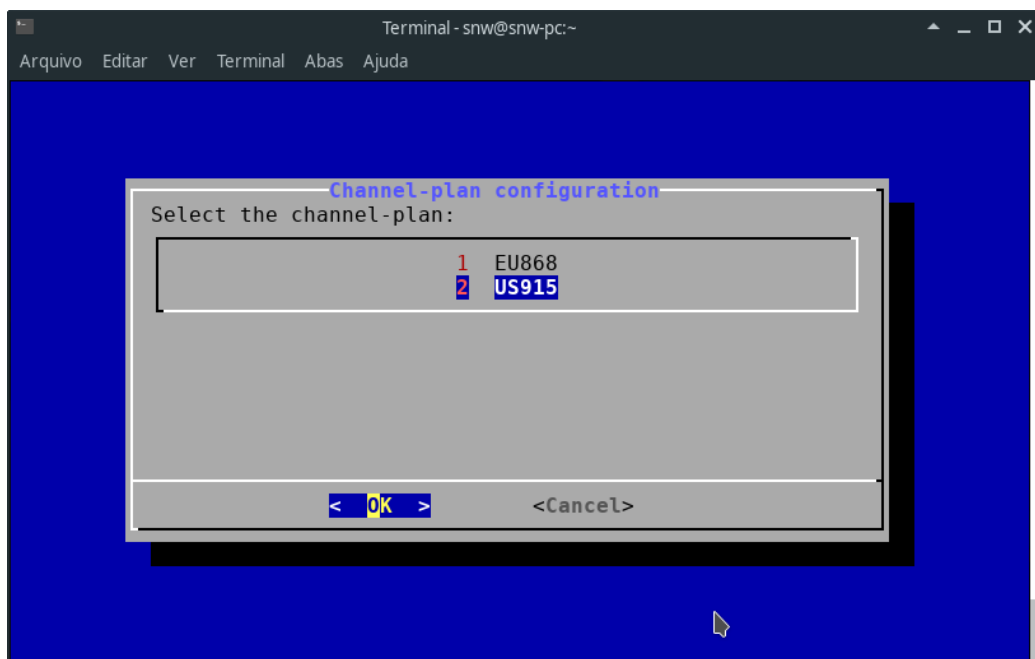
Figura 20 Seleção do módulo concentrador



Fonte: O Autor

Na sequência pede-se a seleção de qual padrão de frequência a ser utilizado, neste caso opção 2 - US915 (Figura 21).

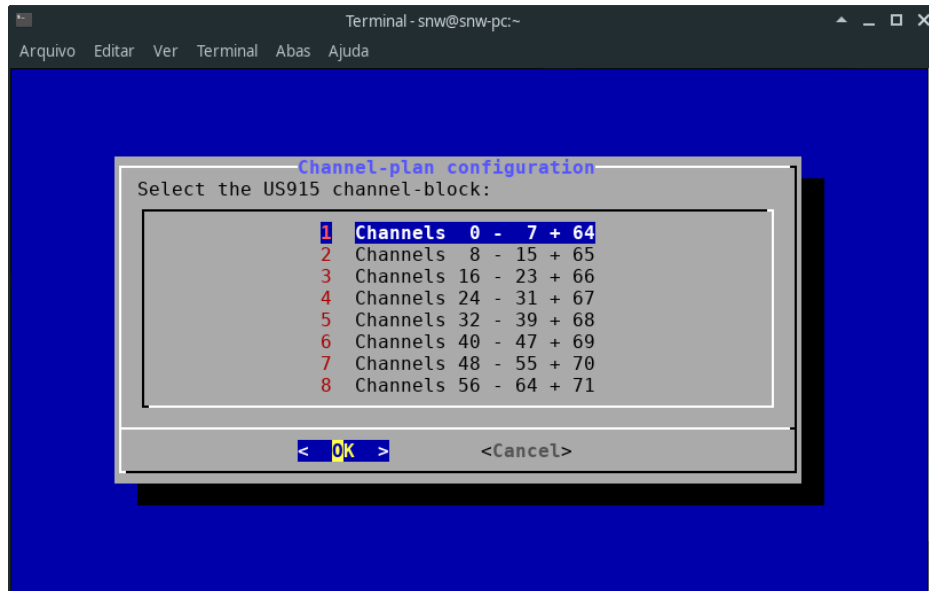
Figura 21 Seleção do padrão de canal



Fonte: O Autor

Por último o bloco de canais selecionado, foi escolhido a opção 1 - canais de 0 a 7 + 64, como na Figura 22, por ser o padrão.

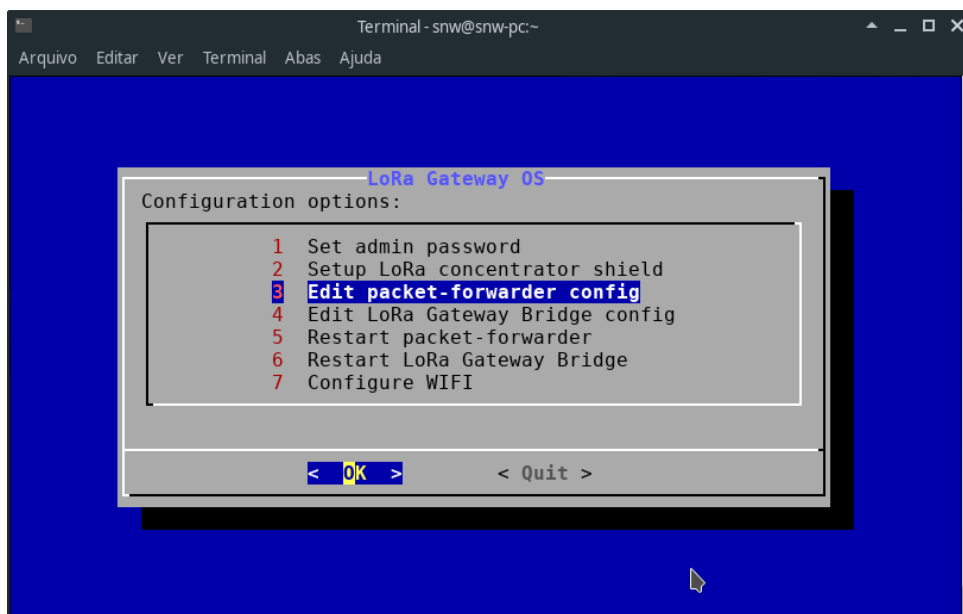
Figura 22 Seleção do bloco de canais



Fonte: O Autor

Dando sequência na configuração do Gateway, retornando para o menu principal, é necessário configurar o packet-forwarder, na opção 3 do menu principal, conforme apresentado na Figura 23.

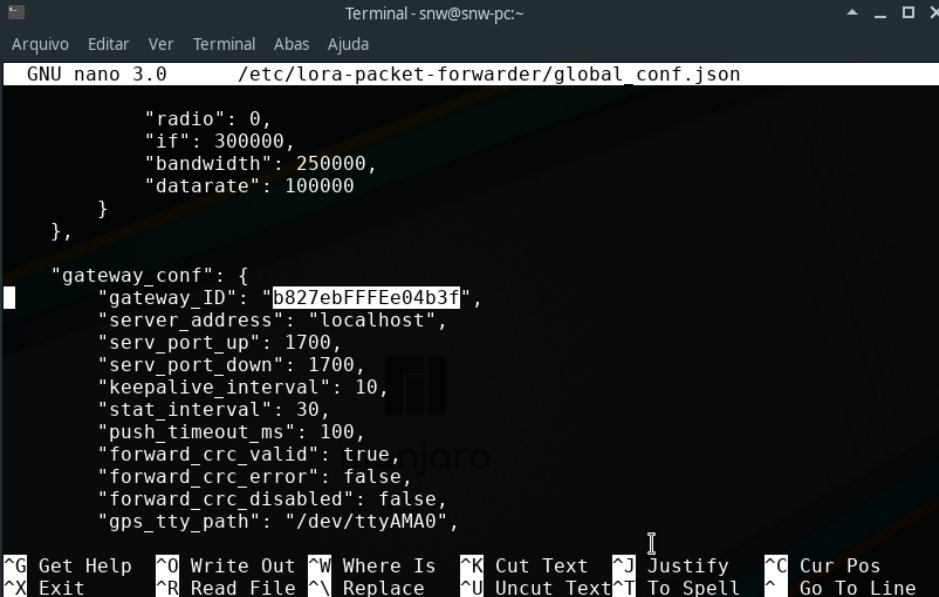
Figura 23 Configuração packet-forwarder



Fonte: O Autor

Esta opção abre um arquivo para edição, o principal ponto a ser configurado neste arquivo é o gateway_ID, dentro de gateway_conf, como mostrado na Figura 24.

Figura 24 Configuração do gateway_ID



```

Terminal - snw@snw-pc:~
Arquivo Editar Ver Terminal Abas Ajuda
GNU nano 3.0 /etc/lora-packet-forwarder/global_conf.json

    "radio": 0,
    "if": 300000,
    "bandwidth": 250000,
    "datarate": 100000
  },
  "gateway_conf": {
    "gateway_ID": "b827ebFFFEe04b3f",
    "server_address": "localhost",
    "serv_port_up": 1700,
    "serv_port_down": 1700,
    "keepalive_interval": 10,
    "stat_interval": 30,
    "push_timeout_ms": 100,
    "forward_crc_valid": true,
    "forward_crc_error": false,
    "forward_crc_disabled": false,
    "gps_tty_path": "/dev/ttyAMA0",
  }
}
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^C Cur Pos
Go To Line

```

Fonte: O Autor

Este ID pode ser gerado na criação do Gateway, através do LoRa App Server, esta criação será abordada na Seção 4.2.

Finalizando as configurações do Gateway, é possível verificar se todos os componentes estão funcionando de acordo, saindo do menu de configuração, na tela inicial do sistema operacional, é listado as seguintes opções de comando: sudo gateway-config - que retorna para o menu citado anteriormente; ou sudo monit status - que mostra o estado dos componentes do sistema.

A seguir na Figura 25 é mostrada a interface do banner inicial do sistema, com a listagem dos comandos.

Figura 25 Tela inicial LoRa Gateway OS

```
Terminal - snw@snw-pc-~
Arquivo Editar Ver Terminal Abas Ajuda
[snw@snw-pc ~]$ ssh admin@192.168.0.25
admin@192.168.0.25's password:
LoRa Gateway OS
documentation and copyright information: www.loraserver.io
Commands:
> sudo gateway-config - configure the gateway
> sudo monit status - display service monitor
raspberrypi3:~$
```

Fonte: O Autor.

Verificando os componentes com *sudo monit status* é possível verificar que todos estão funcionando de acordo nas Figura 26, Figura 27 e Figura 28.

Figura 26 Monitoramento do Status dos módulos do LoRa Server OS.

```
Terminal - snw@snw-pc-~
Arquivo Editar Ver Terminal Abas Ajuda
raspberrypi3:~$ sudo monit status
Password:
Monit 5.25.2 uptime: 2h 45m
Process 'loraserver'
status OK
monitoring status Monitored
monitoring mode active
on reboot start applications / Application_TCC / Devices / Device_teste_TCC
pid 331
parent pid 1
uid 0
effective uid 0
gid 0
uptime 2h 47m
threads 15
children 0
cpu 0.2%
cpu total 0.2%
memory 1.9% [18.1 MB]
memory total 1.9% [18.1 MB]
security attribute -
disk read 0 B/s [14.6 MB total]
data collected Tue, 29 Oct 2019 01:48:45
Process 'lora-packet-forwarder'
status OK
monitoring status Monitored
monitoring mode active
on reboot start
pid 327
parent pid 1
uid 0
effective uid 0
gid 0
uptime 2h 47m
threads 7
```

Fonte: O Autor.

Figura 27 Monitoramento do Status dos módulos do LoRa Server OS.

```

Terminal - snw@snw-pc~
Arquivo Editar Ver Terminal Abas Ajuda
gid 0
uptime 2h 47m
threads 7
children 0
cpu 0.3%
cpu total 0.3%
memory 0.1% [1.3 MB]
memory total 0.1% [1.3 MB]
security attribute -
disk read 0 B/s [118 kB total]
data collected Tue, 29 Oct 2019 01:48:45

Process 'lora-gateway-bridge'
status OK
monitoring status Monitored
monitoring mode active
on reboot start
pid 304
parent pid 1
uid 0
effective uid 0
gid 0
uptime 2h 47m
threads 12
children 0
cpu 0.0%
cpu total 0.0%
memory 1.2% [11.2 MB]
memory total 1.2% [11.2 MB]
security attribute -
disk read 0 B/s [9.1 MB total]
data collected Tue, 29 Oct 2019 01:48:45

Process 'lora-app-server'
status OK
monitoring status Monitored
monitoring mode active
on reboot start

```

Fonte: O Autor.

Figura 28 Monitoramento do Status dos módulos do LoRa Server OS.

```

Terminal - snw@snw-pc~
Arquivo Editar Ver Terminal Abas Ajuda
security attribute -
disk read 0 B/s [9.1 MB total]
data collected Tue, 29 Oct 2019 01:48:45

Process 'lora-app-server'
status OK
monitoring status Monitored
monitoring mode active
on reboot start
pid 299
parent pid 1
uid 0
effective uid 0
gid 0
uptime 2h 47m
threads 15
children 0
cpu 0.5%
cpu total 0.5%
memory 2.8% [25.5 MB]
memory total 2.8% [25.5 MB]
security attribute -
disk read 0 B/s [19.9 MB total]
data collected Tue, 29 Oct 2019 01:48:45

System 'raspberrypi3'
status OK
monitoring status Monitored
monitoring mode active
on reboot start
load average [0.16] [0.13] [0.16]
cpu 3.1%us 1.0%sy 0.0%wa
memory usage 78.9 MB [8.5%]
swap usage 0 B [0.0%]
uptime 2h 47m
boot time Mon, 28 Oct 2019 23:01:27
data collected Tue, 29 Oct 2019 01:48:45

```

Fonte: O Autor.

3.4.2 Acesso ao Gateway e sua adição na rede LoRaWAN

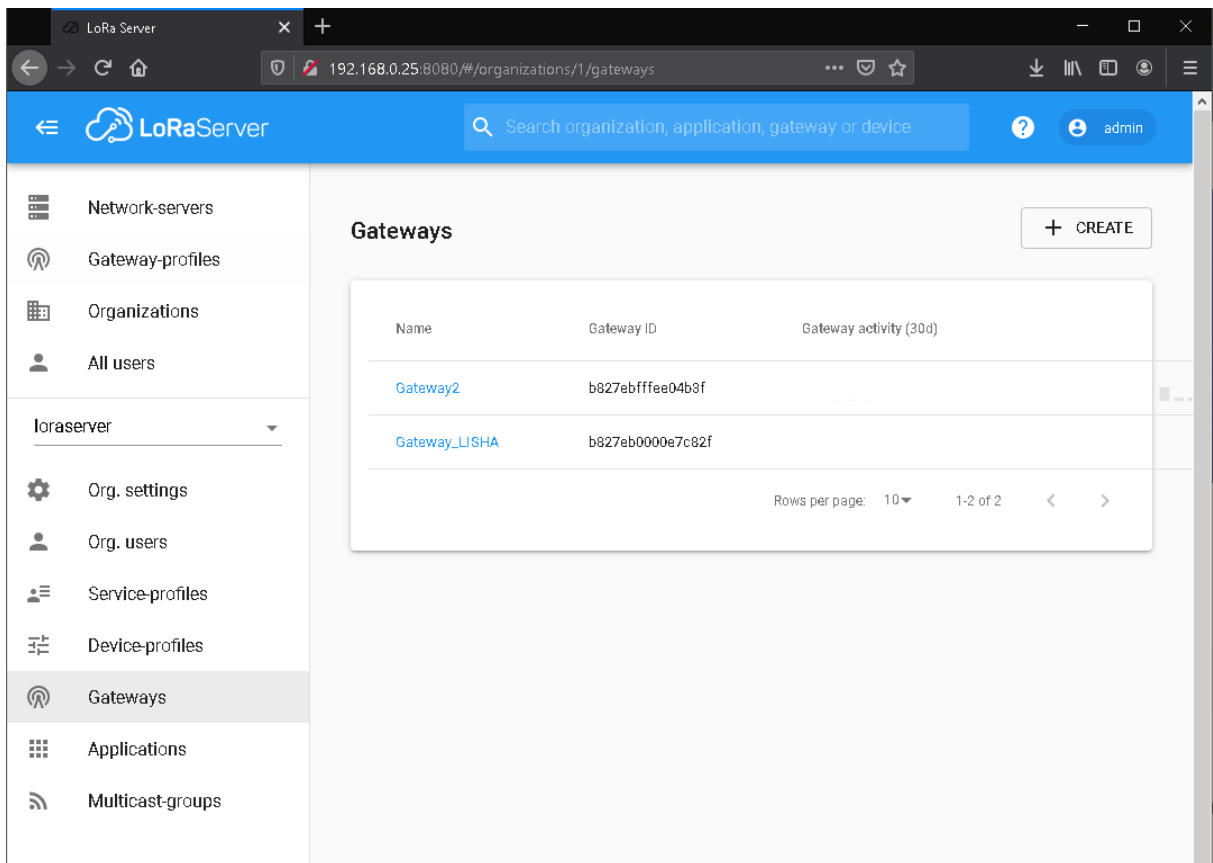
Finalizada a configuração do gateway no Raspberry Pi, pode-se acessar o LoRa App Server através do endereço IP do gateway na porta: 8080 (na configuração padrão, a porta

pode ser alterada no Raspberry Pi). A primeira tela é a de login, com usuário e senha padrão: login: admin e senha: admin. Após o login é necessário criar diversos parâmetros, tais como: Network-server, Gateway-profile e Organização. (SERVER, 2019)

Estes perfis serão necessários para definir a estrutura da rede e os proprietários de cada gateway e dispositivo final cadastrado na rede. Também é possível criar perfis de usuário para a utilização de cada Organização protegida pela permissão de cada usuário.

Partindo para a adição do Gateway na rede, a Figura 29 mostra uma rede com dois gateway cadastrados, para a adição de mais um, deve-se selecionar *Create*.

Figura 29 Gateways disponíveis na rede.



Fonte: O Autor.

Após selecionar a criação, é preciso adicionar um nome para o Gateway, uma descrição e gerar o ID do Gateway, (ID que deve ser configurado no packet-forwarder como visto na seção anterior) e selecionar o Servidor de rede e seu perfil, ambos criados anteriormente. A Figura 30 mostra como ficariam as configurações do novo gateway.

Figura 30 Adição de um gateway na rede

The screenshot shows the LoRa Server web interface for creating a new gateway. The browser address bar shows the URL: 192.168.0.25:8080/#/organizations/1/gateways/create. The page title is 'Gateways / Create'. The form contains the following fields:

- Gateway name ***: Gateway Teste
- Gateway description ***: Gateway adicionado para fins de teste
- Gateway ID ***: b8 27 eb ff fe e0 4b 3f (with a refresh icon)
- Network-server ***: LISHA-LORA
- Gateway-profile**: Gateway1-LISHA
- Gateway discovery enabled

Below the 'Gateway discovery enabled' checkbox, there is a note: "When enabled (and LoRa Server is configured with the gateway discover feature enabled), the gateway will send out periodical pings to test its coverage by other gateways in the same network."

Fonte: O Autor

3.4.3 Criação de um Dispositivo Final na rede e de uma Aplicação

O primeiro passo para adicionar um dispositivo final na rede é criar um Perfil de Dispositivo. No perfil, é necessário adicionar um nome, selecionar a qual servidor de rede ele pertence, sua versão MAC suportada e o poder irradiado efetivo do dispositivo. Uma configuração possível é mostrada na Figura 31. A adição de um dispositivo final ao LoRa Server, configuração e programação para comunicação são necessárias para testar se a rede está devidamente configurada e em funcionamento.

Figura 31 Criação do perfil do dispositivo final.

The screenshot shows the LoRaServer web interface for creating a device profile. The browser address bar shows the URL: 192.168.0.25:8080/#/organizations/1/device-profiles/create. The page title is "Device-profiles / Create". The form is divided into tabs: GENERAL, JOIN (OTAA / ABP), CLASS-B, CLASS-C, and COD. The GENERAL tab is selected and contains the following fields:

- Device-profile name ***: Device_TCC_profile (A name to identify the device-profile.)
- Network-server ***: LISHA-LORA (The network-server on which this device-profile will be provisioned. After creating the device-profile, this value can't be changed.)
- LoRaWAN MAC version ***: 1.1.0 (The LoRaWAN MAC version supported by the device.)
- LoRaWAN Regional Parameters revision ***: A (Revision of the Regional Parameters specification supported by the device.)
- Max EIRP ***: 15 (Maximum EIRP supported by the device.)

A "CREATE DEVICE-PROFILE" button is located at the bottom right of the form.

Fonte: O Autor.

Dando sequência para a próxima aba para a configuração do perfil de dispositivo, é necessário adicionar o delay da primeira janela de downlink do dispositivo final, o offset da taxa de dados, a taxa de dados da segunda janela bem como a frequência do canal a ser utilizado, e a frequência de fábrica dos dispositivos a utilizarem este perfil. Na Figura 32 são demonstrados os valores utilizados nestes parâmetros.

Figura 32 Perfil de dispositivo.

The screenshot shows the LoRaServer web interface for creating a device profile. The browser address bar shows the URL: 192.168.0.25:8080/#/organizations/1/device-profiles/create. The page title is "Device-profiles / Create". The form is divided into tabs: GENERAL, JOIN (OTAA / ABP) (selected), CLASS-B, CLASS-C, and COD. The form contains the following fields and values:

- Device supports OTAA
- RX1 delay * **1**
- RX1 delay (valid values are 0 - 15).
- RX1 data-rate offset* **0**
- Please refer the LoRaWAN Regional Parameters specification for valid values.
- RX2 data-rate* **980**
- Please refer the LoRaWAN Regional Parameters specification for valid values.
- RX2 channel frequency (Hz) * **923300000**
- Factory-preset frequencies (Hz) * **915100000**
- List of factory-preset frequencies (Hz), comma separated.

A "CREATE DEVICE-PROFILE" button is located at the bottom right of the form.

Fonte: O Autor.

O último passo da configuração do dispositivo final é adicioná-lo à rede. Em “Applications” adiciona-se o dispositivo na rede descrevendo seu nome, uma breve descrição e o perfil de serviço. Para a aplicação considerada, não foi selecionado nenhum “Payload codec”. Exemplo na Figura 33.

Figura 33 Criação da Aplicação na rede.

The screenshot shows the LoRaServer web interface for creating a new application. The browser address bar shows the URL: 192.168.0.25:8080/#/organizations/1/applications/create. The page title is 'Applications / Create'. The form contains the following fields:

- Application name*:** Application_TCC. A note below states: 'The name may only contain words, numbers and dashes.'
- Application description*:** Criação do dispositivo na rede para teste.
- Service-profile*:** Service-lisha. A dropdown menu is open showing 'Service-lisha' as the selected option.
- Payload codec:** None. A note below states: 'By defining a payload codec, LoRa App Server can encode and decode the binary device payload for you. Important note: they payload fields have moved to the device-profile. For backward-compatibility and migration, existing codec settings are still visible. Codec settings on the device-profile have priority over the application codec settings.'

A 'CREATE APPLICATION' button is located at the bottom right of the form.

Fonte: O Autor

Depois de criada a aplicação é necessário criar o dispositivo dessa aplicação (Figura 34), permitindo, posteriormente, adicionar alguma integração externa na aplicação, como banco de dados, por exemplo.

Figura 34 Configuração da Aplicação.

The screenshot shows the configuration page for the application 'Application_TCC'. The page has a 'DELETE' button in the top right corner. Below the application name, there are four tabs: 'DEVICES', 'APPLICATION CONFIGURATION', 'INTEGRATIONS', and 'FUOTA'. The 'DEVICES' tab is selected. Below the tabs, there is a '+ CREATE' button. A table with the following columns is shown:

Last seen	Device name	Device EUI	Link margin	Battery

Below the table, there is a 'Rows per page' dropdown set to 10 and a '0-0 of 0' indicator.

Fonte: O Autor.

Na criação do dispositivo, conforme mostrado na Figura 35, inicialmente é adicionado seu nome e descrição, depois precisa-se gerar o EUI, um identificador global único do dispositivo, e selecionar seu perfil. Para fins de teste da rede, optou-se por desabilitar o contador de validação de frames, porém sua utilização pode ajudar a evitar ataques de repetição.

Figura 35 Criação do Dispositivo

Applications / Application_TCC / Devices / Create

GENERAL VARIABLES TAGS

Device name *
Device_teste_TCC

The name may only contain words, numbers and dashes.

Device description *
Criação do dispositivo na rede para teste.

Device EUI *
06 ee c1 22 37 89 f1 dc MSB ↻

Device-profile *
Device_TCC_profile

Disable frame-counter validation

Note that disabling the frame-counter validation will compromise security as it enables people to perform replay-attacks.

CREATE DEVICE

Fonte: O Autor.

Após a criação do dispositivo, seleciona-se a tela de dados de ativação dele, nessa tela, o Endereço do Dispositivo, a Chave de Sessão da Rede e a Chave de Sessão da Aplicação podem ser geradas, e serão utilizadas posteriormente na programação do dispositivo. A seguir um exemplo da configuração na Figura 36.

Figura 36 Configuração de Ativação

The screenshot displays the LoRaServer web interface for configuring a device's activation. The browser address bar shows the URL: `192.168.0.25:8080/#/organizations/1/applications/37/devices/06eec1223789f1dc/activation`. The page title is "Applications / Application_TCC / Devices / Device_teste_TCC". A search bar is present with the text "Search organization, application, gateway or device". The user is logged in as "admin".

The main content area has a breadcrumb trail: "Applications / Application_TCC / Devices / Device_teste_TCC". A "DELETE" button is visible in the top right corner. Below the breadcrumb, there are several tabs: "DETAILS", "CONFIGURATION", "KEYS (OTAA)", "ACTIVATION" (which is selected), "DEVICE DATA", "LORAWAN FRAMES", and "FIRMWARE".

The "ACTIVATION" tab contains the following configuration fields:

- Device address ***: `01 2c 0d ce` (MSB, refresh icon)
- Network session key (LoRaWAN 1.0) ***: `0d af 66 8c 4e 90 b2 a8 fa 2a 46 0c 7b 85 ed 68` (MSB, refresh icon, delete icon)
- Application session key (LoRaWAN 1.0) ***: `39 e6 e1 13 96 7d 8f 56 a9 6d 69 20 42 33 2f cf` (MSB, refresh icon, delete icon)
- Uplink frame-counter***: `0`
- Downlink frame-counter (network) ***: `0`

At the bottom right of the configuration area, there is a "(RE)ACTIVATE DEVICE" button.

Fonte: O Autor.

4 TESTES E RESULTADOS

4.1 SOFTWARE DE TESTE DO DISPOSITIVO FINAL E VALIDAÇÃO DO FUNCIONAMENTO DA REDE LORAWAN

Finalizada a configuração do dispositivo no servidor, é possível realizar a programação do Arduino que irá prover as informações de configuração do dispositivo final, bem como a rotina de envio de dados. Para o uso das funcionalidades do LoRaWAN, foi utilizada a biblioteca LoRaWAN² da Seeed Technology Inc, a qual foi desenvolvida para suportar as funcionalidades LoRaWAN. A biblioteca foi usada como base nesse estudo e foi modificada para atender as demandas dessa aplicação e funcionar corretamente em conjunto com o Arduino MEGA.

O código teste foi baseado em um exemplo contido na pasta adquirida junto a biblioteca LoRaWAN e pode ser visto no Anexo A - deste documento.

4.1.1 Resultados do Teste

Na Figura 37 pode-se observar as respostas do módulo aos comandos AT enviados pelo Arduino, incluindo a versão do firmware do módulo e as novas configurações para a conexão com o servidor. A mesma imagem mostra as mensagens a serem enviadas ao servidor.

² Biblioteca desenvolvida por Wayne Weng, com direitos da Seeed Technology Inc. e licenciamento MIT - direitos de uso, modificação e distribuição liberados pela empresa.

Figura 37 Resposta do Arduino

```

COM6
+VER: 2.1.15
+ID: DevAddr, 01:2C:0D:CE
+ID: DevEui, 06:EE:C1:22:37:89:F1:DC
+ID: AppEui, 06:EE:C1:22:37:89:F1:DC
+ID: DevAddr, 01:2C:0D:CE
+ID: DevEui, 06:EE:C1:22:37:89:F1:DC+KEY: NWKKEY ODAF668C4E90B2A8FA2A460C7B85ED68
+KEY: APPKEY 39E6E113967D8F56A96D692042332FCF
+KEY: APPKEY 39E6E113967D8F56A96D692042332FCF
+MODE: LWAPP
+DR: US915HYBRID
+DR: DR0
+DR: US915HYBRID DR0 SF10 BW125K
+CH: 0,915100000,DR0:DR3
+CH: 1,915300000,DR0:DR3
+CH: 2,915500000,DR0:DR3
+RXWIN1: 0,915100000
+RXWIN2: 916500000,DR8
+LW: DC, OFF, 0
+LW: JDC, OFF
+POWER: 14
+ID: DevAddr, 01:2C:0D:CE
+ID: DevEui, 06:EE:C1:22:37:89:F1:DC
+ID: AppEui, 06:EE:C1:22:37:89:F1:DC
AT+MSDloWorld!"+ID: DevAddr, 01:2C:0D:CE
+ID: DevEui, 06:EE:C1:22:37:89:F1:DC
+ID: AppEui, 06:EE:C1:22:37:89:F1:DC
+MSG: Start
+MSG: Done

```

— Versão do Firmware do módulo LoRa

— Configurações do Dispositivo na Rede

— Enviar a mensagem "HelloWorld!"

— Mensagem Enviada

Fonte: O Autor.

No lado do servidor, a Figura 38 mostra as mensagens sendo recebidas. Pode-se observar no campo “data:” a mensagem “SGVsbG9Xb3JsZCE=”, o servidor recebe ela em base64, decodificando a mensagem em qualquer site decoder de base64 temos a mensagem original “HelloWorld!”. É possível configurar um “payload codec” no servidor de aplicação para que ele decodifique esta mensagem, porém como o objetivo deste teste era apenas verificar a comunicação do dispositivo final com o servidor, neste teste não foi implementado o decodificador, sendo abordado no teste seguinte.

Figura 38 Dados recebidos pelo servidor.

The screenshot displays the LoRaServer web interface. The left sidebar contains navigation options: Network-servers, Gateway-profiles, Organizations, All users, loraserver (selected), Org. settings, Org. users, Service-profiles, Device-profiles, Gateways, Applications, and Multicast-groups. The main content area shows the breadcrumb 'Applications / Application_TCC / Devices / Device_teste_TCC' with a 'DELETE' button. Below this are tabs for DETAILS, CONFIGURATION, KEYS (OTAA), ACTIVATION, and DEVICE DATA (selected). Action buttons include HELP, PAUSE, DOWNLOAD, and CLEAR. The main data area shows a list of uplink messages. The top message is expanded, showing the following JSON data:

```

{
  "adr": true,
  "applicationID": "37",
  "applicationName": "Application_TCC",
  "data": "SGVsbG9uX3JsZCE=",
  "devEUI": "06eec1223789f1dc",
  "deviceName": "Device_teste_TCC",
  "fCnt": 2,
  "fPort": 8,
  "bdinfo": {
    "keys": 2
  },
  "dr": 0,
  "frequency": 903500000
}

```

Below the expanded message, a list of other uplink messages is shown with timestamps and 'uplink' status:

Timestamp	Status
11:05:58 PM	uplink
11:05:46 PM	uplink
11:05:34 PM	uplink
11:05:22 PM	uplink
11:05:10 PM	uplink
11:04:58 PM	uplink
11:04:34 PM	uplink

Fonte: O Autor.

Com este teste pode-se confirmar o funcionamento correto de toda a rede implantada, inclusive a configuração do gateway, do LoRa Server, da comunicação do gateway com o servidor e as informações necessárias para que o Dispositivo Final consiga se comunicar com a rede.

4.2 TESTE DE COBERTURA DE SINAL DENTRO DO PERINI BUSINESS PARK

Visando a possível utilização desta rede implantada, além de validar a área de alcance da tecnologia LoRa, um teste de cobertura dentro do condomínio industrial Perini foi feito, visto que a Universidade Federal de Santa Catarina situa-se no momento dentro do condomínio e é onde está instalado o gateway deste trabalho, no Laboratório de Integração Software/Hardware (LISHA).

O teste não possuiu o objetivo de ser feito com o melhor cenário possível, em um local aberto e com altura para melhorar seu alcance, ao contrário, buscou-se simplesmente

testá-lo com um setup simplista, estando o gateway dentro do LISHA, cercado por paredes, em cima de uma mesa de mais ou menos 70 cm de altura. Além disso, a posição da UFSC dentro do Perini não auxilia com o alcance da área de todo o parque, estando em uma das extremidades do condomínio. Esses pontos representam desafios em atingir o outro lado do parque, visto que existem diversos prédios de indústrias e empresas interferindo no sinal e diminuindo seu alcance.

Para o teste em questão foi desenvolvido uma aplicação no Arduino utilizando além do modem LoRa, um GPS para medir as coordenadas atuais do sistema, mostrando o local de onde o sinal foi emitido e facilitando a medição de distância e mapeamento do alcance.

O programa utilizado no Arduino encontra-se no Apêndice A deste documento.

No início dos testes foi percebido um problema no alcance do sinal, o qual alcançava em torno de 100 m de distância do gateway. Depois de repassar toda a configuração do dispositivo final, rever as opções para melhoria do sinal, foi confirmado que já estava nas melhores condições, restando duas possibilidades para a investigação do problema, ou o módulo do gateway ou o dispositivo final apresentava falhas.

Pensando no dispositivo final como maior chance de erro, visto que o módulo LoRa do dispositivo final estava embarcado em uma placa projetada com fins didáticos, para facilitar as conexões com o módulo, surgiu a hipótese de haver um problema na trilha de ligação entre o módulo e a antena. Para confirmar a hipótese foram realizadas medições de intensidade de sinal com e sem a antena, a informação da intensidade de sinal pode ser visualizada no pacote de Uplink recebido pelo gateway e mostrada em RSSI (Indicador de intensidade do sinal recebido, no LoRa App Server).

No teste em questão foi percebido que não havia diferença entre a intensidade de sinal enviada com ou sem a antena, fazendo com que esta hipótese se torne mais forte. Após a realização dos testes, a antena foi soldada no módulo LoRa, ignorando a trilha da placa em que se encontrava.

O ganho no sinal foi notável e o sinal que antes mesmo o modem estando a centímetros de distância do gateway, era em média de -65 dBm, passou a ter uma média de -12 dBm, o que significa que o sinal que chegava com praticamente 50% de intensidade passou a chegar com mais de 90%.

Após a mudança foi feito o teste do GPS e do programa Arduino citado anteriormente, também foi feita a integração do servidor com o InfluxDB para que os dados fossem salvos nele, bem como também foi adicionado um “Payload codec”, função do LoRa App Server para tratar os dados do payload recebido, este codec foi feito em JavaScript e

converte os dados recebidos em string para hexadecimal, em sequência são tratados para as coordenadas fiquem mais fáceis de serem lidas.

O “payload codec” foi adicionado no Apêndice B deste documento, nele também está incluso o tratamento de mensagens que não são coordenadas, sendo neste caso os bytes transformados em ASCII. Com os dados das coordenadas adquiridos, foi criado um mapa de onde a mensagem foi enviada, trazendo uma melhor visualização das distâncias alcançadas, o mapa pode ser visto na Figura 39.

Figura 39 Locais de onde houve comunicação entre dispositivo final e gateway.



Fonte: O Autor – Através da ferramenta MyMaps do Google.

4.2.1 Resultados do Teste

Com a coleta destes dados pode-se observar um alcance de até cerca de 800 m de linha reta, porém como a área do condomínio é cheia de prédios e árvores, o sinal não é homogêneo, fazendo com que esse alcance varie de acordo com a quantidade e tamanho dos prédios entre o dispositivo final e o gateway.

De um modo geral como trata-se de uma área não só urbana, mas também industrial, podendo conter diversas fontes de ruído além de prédios com diversas paredes e máquinas, o módulo RHF76-052 se saiu bem com seus 14 dBm de potência de sinal no envio para

915Mhz. Para melhores resultados em questão de distância, o ideal seria a utilização de algum módulo que suporte uma maior potência máxima de saída, como o módulo RF LoRa1276, baseado no chip SX1276 da Semtech, que possui uma potência máxima de 20 dBm.

5 CONCLUSÃO

Conforme apresentado neste trabalho, toda a implantação da rede LoRa foi feita buscando simplicidade, não tão trivial de se fazer sem conhecimento, porém após o aprendizado sobre o funcionamento da rede e sua arquitetura, é algo simples de ser implementado. Como o objetivo deste trabalho inclui a análise da implantação da rede, pode-se tomar algumas notas sobre as vantagens e desvantagens de ser dono do próprio servidor de rede, versus a utilização de um servidor externo como por exemplo o The Things Network, que possui cobertura em diversos países.

Existem dois casos possíveis para essa análise, a existência ou não de sinal de um gateway de servidor externo na região a ser utilizada.

Caso exista cobertura de pelo menos um gateway, há uma vantagem clara que é não necessitar comprar um gateway e nem configura-lo na rede, o que simplifica o trabalho quase que pela metade.

Porém ainda não é tão fácil encontrar tal cobertura em muitas cidades e onde há cobertura, ela não atinge toda a cidade ou área rural. Neste caso, a dificuldade e gasto se tornam muito parecidos, tendo que ser comprado o gateway e configurado para se comunicar com o servidor externo.

Entre desvantagens de utilizar um servidor externo está, conforme citado na seção 2.4.2 em AppSKey, no fato do servidor de rede apesar de ser considerado confiável, ainda pode alterar o conteúdo da mensagem em trânsito. Desta maneira caso o usuário deseje uma maior segurança, ele teria que criptografar sua mensagem e lidar com todo o tratamento desta mensagem pós servidor, o que adiciona uma complexidade que muitas vezes exige conhecimento desconhecido para usuários normais.

Já no caso de ser dono do próprio servidor, o usuário não tem a preocupação com a segurança dos seus dados no caso de poderem ser alterados pelo servidor, restando a preocupação com a segurança de acesso, que em ambos os casos é a mesma, a das credenciais de login.

Falando sobre os resultados da rede implantada neste trabalho, obteve-se um resultado satisfatório, pois além do correto funcionamento da rede e comunicação entre dispositivo final - gateway - servidor, obteve-se também um alcance aceitável de comunicação para a área onde foi realizado o teste e pela potência do sinal do módulo utilizado.

Assim os objetivos deste trabalho foram cumpridos, a rede foi implantada e testada com sucesso. Além disso foi feito um teste para verificar a distância alcançada entre os módulos utilizados.

Como o objetivo do trabalho foi alcançado, indica-se como possível trabalho futuro a utilização e análise de outras classes de dispositivos para a rede LoRa, bem como uma análise de consumo de energia entre as classes.

REFERÊNCIAS

- ANGELIS, Guido de et al. **A hybrid outdoor/indoor Positioning System for IoT applications**. 2015 Ieee International Symposium On Systems Engineering (isse), [s.l.], v. 1, n. 1, p.1-6, set. 2015. IEEE. <http://dx.doi.org/10.1109/syseng.2015.7302503>.
- ARDUINO. **ARDUINO MEGA 2560 REV3**. 2019. Disponível em: <<https://store.arduino.cc/usa/mega-2560-r3>>. Acesso em: 22 out. 2019.
- Augustin, A.; Yi, J.; Clausen, T.; Townsley, W.M. **A Study of LoRa: Long Range & Low Power Networks for the Internet of Things**. Sensors 2016, 16, 1466. Acesso em: 17 jun. 2018.
- BABGEI, Atar. **RisingHF RHF0M301 - Raspberry Pi Adapter**. 2017. Disponível em: <<https://github.com/ttn-surabaya/RHF0M301-Pi-Adapter>>. Acesso em: 12 ago. 2019.
- BRUYNE, Jeffrey de et al. **Comparison of the Link Budget with Experimental Performance of a WiMAX System**. Eurasip Journal On Wireless Communications And Networking, [s.l.], v. 2009, n. 1, p.1-8, dez. 2009. Springer Science and Business Media LLC. <http://dx.doi.org/10.1155/2009/247436>.
- CENTENARO, Marco et al. **Long-range communications in unlicensed bands: the rising stars in the IoT and smart city scenarios**. Ieee Wireless Communications, [s.l.], v. 23, n. 5, p.60-67, out. 2016. Institute of Electrical and Electronics Engineers (IEEE). <http://dx.doi.org/10.1109/mwc.2016.7721743>.
- CHIRPSTACK. **ChirpStack, open-source LoRaWAN® Network Server stack**. Disponível em: <<https://www.chirpstack.io/>>. Acesso em: 2 jul. 2019.
- COMMITTEE, **Lora Alliance Technical**. LoRaWAN™ 1.0.3 Specification. 2018. Disponível em: <<https://lora-alliance.org/sites/default/files/2018-07/lorawan1.0.3.pdf>>. Acesso em: 26 ago. 2019..
- COSTA, Cainã L.; OLIVEIRA, Letícia; MÓTA, Léia Michele S.. **Internet das coisas (IOT): um estudo exploratório em agronegócios**. In: VI SIMPÓSIO DA CIÊNCIA DO AGRONEGÓCIO, 1., 2018, Porto Alegre. [S.I.]. [s.i.]: [s.i.], 2018. p. 1 - 10.
- EDITORS, Digi-key's European. **Ensuring Device and Radio Security in Low-Power IoT Wireless Connections**. 2017. Disponível em: <<https://www.digikey.com/en/articles/techzone/2017/aug/ensuring-device-radio-security-lorawan>>. Acesso em: 29 out. 2019
- FEHRI, C. et al. **LoRa technology MAC layer operations and Research issues**. Procedia Computer Science, v. 130, p.1096-1101, 2018. Disponível em: <<http://dx.doi.org/10.1016/j.procs.2018.04.162>>. Acesso em: 23 abr. 2018.
- FOUNDATION, **Raspberry Pi. Raspberry Pi 2 Model B**. Disponível em: <<https://www.raspberrypi.org/products/raspberry-pi-2-model-b/>>. Acesso em: 6 ago. 2019.
- H Aidine, Abdelfatteh et al. **The Role of Communication Technologies in Building Future Smart Cities**. Smart Cities Technologies, [s.l.], v. 1, n. 1, p.1-24, 7 dez. 2016. InTech.

<http://dx.doi.org/10.5772/64732>.

INFLUXDATA. **InfluxDB 1.x**. Disponível em: <<https://www.influxdata.com/time-series-platform/>>. Acesso em: 17 set. 2019.

LEE, In; LEE, Kyoochun. **The Internet of Things (IoT): Applications, investments, and challenges for enterprises**. Business Horizons, v. 58, n. 4, p.431-440, jul. 2015. Disponível em: <<http://dx.doi.org/10.1016/j.bushor.2015.03.008>>. Acesso em: 23 abr. 2018.

LIAGKOU, Vasiliki; STYLIOS, Chrysostomos; SALMAS, Dimitrios. **VR training model for exploiting security in LPWAN**. Procedia Cirp, [s.l.], v. 79, n. 1, p.724-729, jan. 2019. Elsevier BV. <http://dx.doi.org/10.1016/j.procir.2019.02.022>.

LORA-ALLIANCE. LoRaWAN™ What is it?: **A technical overview of LoRa® and LoRaWAN™**. 2015. Disponível em: <<https://lora-alliance.org/sites/default/files/2018-04/what-is-lorawan.pdf>>. Acesso em: 17 jun. 2018

LORA-ALLIANCE. **What is the LoRaWAN® Specification?** 2019. Disponível em: <<https://lora-alliance.org/about-lorawan>>. Acesso em: 2 set. 2019.

MARQUES, Julio José Branquinho; BOCHIE, Kaylani. **LoRa**. Disponível em: <<https://www.gta.ufrj.br/ensino/eel878/redes1-2018-1/trabalhos-vf/lora/>>. Acesso em: 15 out. 2019.

MEKKI, Kais et al. **A comparative study of LPWAN technologies for large-scale IoT deployment**. Ict Express, [s.l.], v. 5, n. 1, p.1-7, mar. 2019. Elsevier BV. <http://dx.doi.org/10.1016/j.ict.2017.12.005>.

NOKIA. LTE-M – **Optimizing LTE for the Internet of Things While Paper**. 2015. Disponível em: < <https://novotech.com/docs/default-source/default-document-library/lte-m-optimizing-lte-for-the-internet-of-things.pdf?sfvrsn=0> >. Acesso em: 16 jun. 2018

RASPBIAN. **About Raspbian**. Disponível em: <<https://www.raspbian.org/RaspbianAbout>>. Acesso em: 1 ago. 2019.

RF, Loja. **RHF76-052 / RHF78-052 LoRa Module**. 2018. Disponível em: <<http://www.lojarf.com/risinghf/wireless-module/rhf76-052-rhf78-052-lora-module>>. Acesso em: 17 jun. 2018.

RIÇATO, Filipe Travizani. **Proposta de um Sistema de Monitoramento em Tempo Real, para Estudos de Determinação de Economia de Energia, em Aquecimento Solar de Água**. 2018. 82 f. TCC (Graduação) - Curso de Engenharia Elétrica, Centro de Tecnologia e Urbanismo, Universidade Estadual de Londrina, Londrina, 2018.

RISINGHF. **LoRaWAN AT Command Specification V3.1**. 2016. Disponível em: <<https://www.thethingsnetwork.org/forum/uploads/default/original/2X/9/9b9067781ac614cff3d2e25b669c084e09e47751.pdf>>. Acesso em: 12 ago. 2019.

RISINGHF. **LoRaWAN Module RHF76-052 User Manual**. 2016. Disponível em: <<https://fccid.io/2AJUZ76052/User-Manual/Users-Manual-3211050>>. Acesso em: 17 jun. 2018.

RISINGHF. **RHF0M301 Datasheet V1.2.** Disponível em: <<https://www.robotshop.com/media/files/pdf/915mhz-lora-gateway-raspberry-pi-hat-datasheet1.pdf>>. Acesso em: 6 ago. 2019.

SANTOS, Bruno P. et al. **Internet das Coisas: da Teoria à Prática.** 2107. Disponível em: <<https://homepages.dcc.ufmg.br/~mmvieira/cc/papers/internet-das-coisas.pdf>>. Acesso em: 04 set. 2019.

SEMTECH. **AN120.22 LoRa Modulation Basics.** [S.l.], 2015.

SEMTECH. **What is LoRa?** Disponível em: <<https://www.semtech.com/lora/what-is-lora>>. Acesso em: 19 ago. 2019.

SERVER, Lora. **Adaptive data-rate.** Disponível em: <<https://www.loraserver.io/network-server/features/adaptive-data-rate/>>. Acesso em: 15 jul. 2019.

SERVER, Lora. **Architecture.** Disponível em: <<https://github.com/brocaar/loraserver>>. Acesso em: 15 jul. 2019.

TEIXEIRA, Grazielle Bonaldi; ALMEIDA, João Vítor Peroni de. **REDE LORA® E PROTOCOLO LORAWAN® APLICADOS NA AGRICULTURA DE PRECISÃO NO BRASIL.** 2017. 77 f. TCC (Graduação) - Curso de Engenharia Eletrônica, Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2017.

U-BLOX. **NEO-6 u-blox 6 GPS Modules: Data Sheet.** Disponível em: <[https://s3-sa-east-1.amazonaws.com/robocore-lojavirtual/976/NEO-6_DataSheet_\(GPS.G6-HW-09005\).pdf](https://s3-sa-east-1.amazonaws.com/robocore-lojavirtual/976/NEO-6_DataSheet_(GPS.G6-HW-09005).pdf)>. Acesso em: 21 out. 2019.

Vangelista L., Zanella A., Zorzi M. **Long-Range IoT Technologies: The Dawn of LoRa™.** In: Atanasovski V., Leon-Garcia A. (eds) Future Access Enablers for Ubiquitous and Intelligent Infrastructures. vol 159 Springer, pp. 51-58, 2015

VANGELISTA, Lorenzo; ZANELLA, Andrea; ZORZI, Michele. **Long-Range IoT Technologies: The Dawn of LoRa™.** Lecture Notes Of The Institute For Computer Sciences, Social Informatics And Telecommunications Engineering, [s.l.], p.51-58, 2015. Springer International Publishing. http://dx.doi.org/10.1007/978-3-319-27072-2_7.

YUAN, Michael. **Getting to know MQTT.** 2017. Disponível em: <<https://developer.ibm.com/articles/iot-mqtt-why-good-for-iot/>>. Acesso em: 3 set. 2019.

B, Eric. **LoRa.** Disponível em: <<https://lora.readthedocs.io/en/latest/>>. Acesso em: 22 out. 2019.

ARDUINO. **Arduino Mega 2560 Rev3.** Disponível em: <<https://store.arduino.cc/usa/mega-2560-r3>>. Acesso em: 22 out. 2019.

RisingHF. **AN01541: Set up LoRaWAN GW with RHF0M301.** Disponível em: <https://github.com/SeeedDocument/forum_doc/raw/master/reg/%5BRHF-AN01541%5DSet%20up%20LoRaWAN%20GW%20with%20RHF0M301_v1.8.pdf>. Acesso em: 17 jun. 2018.

APÊNDICE A

```

#include <LoRaWan.h>
#include <TinyGPS.h>
#include <SoftwareSerial.h>
#define SerialGPS Serial1

TinyGPS GPS;
unsigned char data[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 0xA,};
char buffer[256];

static float lat, lon, vel;
static unsigned long day, hora;
static unsigned short sat;
int count = 0;

void setup(void)
{
    SerialGPS.begin(9600);
    SerialUSB.begin(115200);
    while(!SerialUSB);

    lora.init();
    memset(buffer, 0, 256);
    lora.getVersion(buffer, 256, 1);
    SerialUSB.print(buffer);

    memset(buffer, 0, 256);
    lora.getId(buffer, 256, 1);
    SerialUSB.print(buffer);
    lora.setId("00 fb 3f bf", "669deae72a18671c", "669deae72a18671c");
    lora.setKey("9d 96 3d 72 1f aa 67 f5 db 9e cb a3 99 ec cd 9e", "6f 0d 90 a8 99 a7 ca bc 5e e9 41 33 89 0b 72
c7", "6f 0d 90 a8 99 a7 ca bc 5e e9 41 33 89 0b 72 c7");
    lora.setDeciveMode(LWABP);
    lora.setDataRate(DR0, US915HYBRID);
    lora.setChannel(0, 915.1);
    lora.setChannel(1, 915.3);
    lora.setChannel(2, 915.5);

    lora.setReceiceWindowFirst(0, 915.1);
    lora.setReceiceWindowSecond(916.5, DR3);
    lora.setDutyCycle(false);
    lora.setJoinDutyCycle(false);
    lora.setPower(14);
}

void loop(void)
{
    while(SerialGPS.available())
    {
        if (GPS.encode(SerialGPS.read())) {
            //Hora e data
            GPS.get_datetime(&day, &hora);

            //latitude e longitude
            GPS.f_get_position(&lat, &lon);

            SerialUSB.println("TESTEEE");
            //velocidade
            vel = GPS.f_speed_kmph();
        }
    }
}

```

```

        //Satellites
        sat = GPS.satellites();
        //Serial.print("GET!!");
        if (sat != TinyGPS::GPS_INVALID_SATELLITES)
        {
            Serial.print("Satellites: ");
            Serial.println(sat);
        }

        if(count == 2)
        {
            sendmessage();
            count = 0;
        }
        count++;
    }
}

void sendmessage()
{
    //unsigned char data[10] = {'K', '_', '_', '_', '_', '_', '_', '_', '_', '_'};
    data[0] = 'K';
    bool result = false;

    memset(buffer, 0, 256);
    lora.getId(buffer, 256, 1);
    Serial.print(buffer);

    GPS_MSG(data);
    Serial.println(data[0]);
    Serial.println(data[1]);
    Serial.println(data[2]);
    Serial.println(data[3]);
    Serial.println(data[4]);
    Serial.println(data[5]);
    Serial.println(data[6]);
    Serial.println(data[7]);
    Serial.println(data[8]);
    Serial.println(data[9]);

    result = lora.transferPacket(data, 10, 10);

    SerialUSB.print("result is: ");
    SerialUSB.println(result);
    SerialUSB.println(buffer);

    if(result)
    {
        short length;
        short rssi;
        memset(buffer, 0, 256);
        length = lora.receivePacket(buffer, 256, &rssi);

        if(length)
        {
            SerialUSB.print("Length is: ");
            SerialUSB.println(length);
            SerialUSB.print("RSSI is: ");
            SerialUSB.println(rssi);
            SerialUSB.print("Data is: ");
            for(unsigned char i = 0; i < length; i++)
            {

```

```

        SerialUSB.print("0x");
        SerialUSB.print(buffer[i], HEX);
        SerialUSB.print(" ");
    }
    SerialUSB.println();
}
}
}

```

```

void GPS_MSG(unsigned char * data)
{
    uint32_t lat2, lon2;
    lat2 = 0;
    lon2 = 0;

    Serial.println(lat, 6);
    Serial.println(lon, 6);
    data[0] = (lat < 0) ? '-' : '+';
    data[5] = (lon < 0) ? '-' : '+';

    data[1] = getPos((int)lat);
    data[2] = ',';
    lat2 = getPos((int)(lat*100));
    data[3] = lat2 % 100;
    lat2 = getPos((int)((lat - (int)lat)*10000));
    data[4] = lat2 % 100;

    data[6] = getPos((int)lon);
    data[7] = ',';
    lon2 = getPos((int)(lon*100));
    data[8] = lon2 % 100;
    lon2 = getPos((int)((lon - (int)lon)*10000));
    data[9] = lon2 % 100;
}

```

```

int getPos(int num){
    if(num<0){
        num = num*(-1);
    }
    return num;
}

```

APÊNDICE B

Payload Codec utilizado no LoRa App Server

```

function hex_to_ascii(str1)
{
    var hex = str1.toString();
    var str = "";
    for (var n = 0; n < hex.length; n += 2) {
        str += String.fromCharCode(parseInt(hex.substr(n, 2), 16));
    }
    return str;
}

function toHexString(bytes) {
    return bytes.map(function(byte) {
        return ("00" + (byte & 0xFF).toString(16)).slice(-2)
    }).join("")
}

function get_number(i) {
    var str = i.toString();
    if(str.length == 1) str = '0' + str;
    return str;
}

function get_pos(start, tohex) {
    var signal = String.fromCharCode(parseInt(tohex.substr(start, 2), 16))

    if(signal != '+' && signal != '-'){
        return null;
    }
    var vet = []
    var i = 0;
    for(i = 0; i < 4; i++) {
        vet.push(parseInt(tohex.substr(start + 2 + 2*i, 2), 16))
    }
    var str = signal + vet[0].toString() + '.' + get_number(vet[2]) + get_number(vet[3])
    return str
}

function Decode(fPort, bytes) {
    var tohex = toHexString(bytes);
    var toascii = hex_to_ascii(tohex);
    var lat = get_pos(0,tohex);
    if(lat == null){
        return {"mydata": toascii};
    }
    var lon = get_pos(10,tohex)

    return {"lat_lon": (lat + ' ' + lon) };
}

```

ANEXO A

Código para teste do funcionamento da rede:

```
// -- BIBLIOTECA USADA COMO BASE
#include <LoRaWan.h>

unsigned char data[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 0xA,};
char buffer[256];

void setup(void)
{
    SerialUSB.begin(115200);
    while(!SerialUSB);

    lora.init();

    memset(buffer, 0, 256);
    lora.getVersion(buffer, 256, 1);
    SerialUSB.print(buffer);

    memset(buffer, 0, 256);
    lora.getId(buffer, 256, 1);
    SerialUSB.print(buffer);

    //---- setId(char *DevAddr, char *DevEUI, char *AppEUI)
    lora.setId("01 2c 0d ce", "06eec1223789f1dc", "06eec1223789f1dc");

    //---- void LoRaWanClass::setKey(char *NwkSKey, char *AppSKey, char *AppKey)
    lora.setKey("0d af 66 8c 4e 90 b2 a8 fa 2a 46 0c 7b 85 ed 68", "39 e6 e1 13 96 7d 8f 56 a9 6d 69 20 42 33 2f
cf", "39 e6 e1 13 96 7d 8f 56 a9 6d 69 20 42 33 2f cf");

    lora.setDeciveMode(LWABP);
    lora.setDataRate(DR0, US915HYBRID);

    lora.setChannel(0, 915.1);
    lora.setChannel(1, 915.3);
    lora.setChannel(2, 915.5);

    lora.setReceiceWindowFirst(0, 915.1);
    lora.setReceiceWindowSecond(916.5, DR3);

    lora.setDutyCycle(false);
    lora.setJoinDutyCycle(false);

    lora.setPower(14);
}

void loop(void)
{
    bool result = false;

    memset(buffer, 0, 256);
    lora.getId(buffer, 256, 1);
    Serial.print(buffer);

    // Envio da mensagem HelloWorld! para o servidor
    result = lora.transferPacket("HelloWorld!", 11);
}
```

```
if(result)
{
    short length;
    short rssi;

    memset(buffer, 0, 256);
    length = lora.receivePacket(buffer, 256, &rssi);

    if(length)
    {
        SerialUSB.print("Length is: ");
        SerialUSB.println(length);
        SerialUSB.print("RSSI is: ");
        SerialUSB.println(rssi);
        SerialUSB.print("Data is: ");
        for(unsigned char i = 0; i < length; i++)
        {
            SerialUSB.print("0x");
            SerialUSB.print(buffer[i], HEX);
            SerialUSB.print(" ");
        }
        SerialUSB.println();
    }
}
}
```