UNIVERSIDADE DO VALE DO RIO DOS SINOS - UNISINOS
UNIDADE ACADÊMICA DE GRADUAÇÃO
CURSO DE ENGENHARIA ELÉTRICA

LEANDRO GABRIEL PIOVESAN

UTILIZAÇÃO DE REDES LPWAN PARA GERENCIAMENTO DE DISPOSITIVOS *SMART GRID* USANDO O PROTOCOLO DNP3

São Leopoldo
2019

LEANDRO GABRIEL PIOVESAN

UTILIZAÇÃO DE REDES LPWAN PARA GERENCIAMENTO DE DISPOSITIVOS *SMART GRID* USANDO O PROTOCOLO DNP3

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do título de Bacharel em Engenharia Elétrica, pelo Curso de Graduação da Universidade do Vale do Rio dos Sinos - UNISINOS

Orientador: Prof. Ms. Lúcio Renê Prade

São Leopoldo

2019

Aos meus pais Luiz e Lúcia, por terem me gerado, aos meus irmãos André e William, pelas palavras de incentivo, a minha esposa Patrícia, pela paciência e apoio incondicional durante todos esses anos e a minha filha Giovanna, pelos seus belos sorrisos motivadores que, com certeza, valem muito mais do que barras de ouro.

AGRADECIMENTOS

Muitas foram as pessoas que passaram pelo meu caminho e que, de alguma forma, contribuíram para este momento. Portanto, começo agradecendo a todos que, de forma direta ou indireta, me ensinaram algo durante minha formação acadêmica.

As pessoas que contribuíram diretamente para que este trabalho se concretizasse ...

... a minha esposa Patrícia Fischer Barbosa, pelas inúmeras revisões textuais que realizou desde o início e por atender as necessidades da nossa filha enquanto eu escrevia esta monografia.

... ao meu irmão e Eng. Eletricista, André Luiz Piovesan, pelas inúmeras revisões realizadas nesta monografia.

... ao meu pai Luiz Piovesan, pelos auxílios em testes realizados em campo.

... aos meus amigos, colegas de trabalho e engenheiros, Alexandre Felin Gindri, pelas conversas e dicas para o desenvolvimento dos códigos fontes envolvidos e Diogo Koenig, pelas conversas relacionadas a antenas, rádios, potência e atenuação.

... ao meu colega de graduação e agora amigo, Gustavo Thomé, pelas conversas referentes ao presente trabalho, de modo a elucidar problemas e encontrar soluções de contorno.

... ao meu orientador, MS. Lúcio Renê Prade, pela disponibilidade e ajuda ao longo do desenvolvimento deste trabalho.

RESUMO

Este trabalho de conclusão de curso propõe uma outra maneira de acesso para gerenciar remotamente equipamentos que utilizem o protocolo DNP3 (*Distributed Network Protocol version 3*) pois, com o advento e grande utilização de Redes Inteligentes (*smart grids*), tem-se utilizado, cada vez mais, este protocolo para leitura de status e envio de comandos em diversos setores da elétrica, desde a geração, passando pela transmissão até o consumidor, sendo que isto possibilita a redução de custos e grande agilidade na investigação de falhas em redes elétricas. Atualmente, muitas das redes utilizadas para estes fins acabam sendo as redes de telefonia móvel, mas ao pensarmos na quantidade de medidores em residências, comércios e indústrias, começa a ficar inviável a monitoração utilizando este tipo de rede. Com o avançar da Internet das Coisas (IoT – *Internet of Things*), muitos meios de comunicação surgiram e, entre eles, o que este trabalho tem como objetivo explorar, que são as redes de longa distância utilizando tecnologia LoRa, as quais prometem baixo consumo de energia, o que possibilita a utilização com baterias e, o mais importante, sem a utilização de fios. Com isso, pode-se tornar viável o monitoramento elétrico e este meio poderá ser expandido para utilização em outras necessidades onde haja a necessidade de monitoramento de variáveis, como, por exemplo, em setores de água, gás e mobilidade urbana. Após a realização do desenvolvimento de *software* para o módulo de rádio LoRa, observou-se que a utilização deste pode ser viável, sendo necessário ajustes relacionados a temporização (*timeout*), para no mínimo 20 segundos, entre a solicitação e a resposta, quando utiliza-se comandos de 292 *bytes* de dados, neste caso ocorrerá a divisão do pacote em 3 partes, transmitidas com intervalos de até 5 segundos, deste modo, atende-se a especificação DNP3-L1 e parcialmente a especificação DNP3-L2. Convém ressaltar que, em sistemas de gerenciamento que exijam grande disponibilidade e confiabilidade nas comunicações, algumas destas limitações precisam ser analisadas cuidadosamente, pois como se trata de um meio de transmissão, a perda de informações pode ocasionar grandes problemas.

Palavras-chave: Internet das Coisas. *Internet of Things*. Redes Inteligentes. *Smart Grids. DNP3. LPWAN. LoRaWAN. LoRa.*

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE SIGLAS

2G	*Second-generation cellular technology* (Tecnologia celular de segunda geração)
3G	*Third-generation cellular technology* (Tecnologia celular de terceira geração)
4G	*Fourth-generation cellular technology* (Tecnologia celular de quarta geração)
5G	*Fifth-generation cellular technology* (Tecnologia celular de quinta geração)
6LoWPAN	*IPv6 over Low Power Wireless Personal Area Networks* (Redes de área pessoal sem fio IPv6 sobre baixa potência)
A	Ampere
ABNT	Associação Brasileira de Normas Técnicas
ABP	*Activation by Personalization* (Ativação através de Personalização)
ACK	*Acknowledgement* (Confirmação)
AES	*Advanced Encryption Standard* (Padrão Avançado de Criptografia)
AM	*Amplitude Modulation* (Amplitude Modulada)
ASCII	*American Standard Code for Information Interchange* (Código Padrão Americano para Intercâmbio de Informações)
B/s	*Bytes* por segundo
Bit	*Binary digit* (Digito binário)
BLE	*Bluetooth Low Energy* (*Bluetooth* de baixo consumo de energia)
Cm	Centímetros
CRC	*Cyclic Redundancy Check* (Verificação cíclica de redundância)
CSS	*Chirp Spread Spectrum* (Trinados de Modulação de Espectro espalhado)
dB	*Decibel*
dBm	*decibel miliwatt*
DC	*Direct current* (Corrente Contínua)
DNP3	*Distributed Network Protocol version 3* (Protocolo de rede distribuída versão 3)

DNP3-L1	*Distributed Network Protocol version 3 – Level 1* (Protocolo de rede distribuída versão 3 – Nível 1)
DNP3-L2	*Distributed Network Protocol version 3 – Level 2* (Protocolo de rede distribuída versão 3 – Nível 2)
DNP3-L3	*Distributed Network Protocol version 3 – Level 3* (Protocolo de rede distribuída versão 3 – Nível 3)
EDGE	*Enhanced Data Rates for GSM Evolution* (Taxas de dados aprimoradas para evolução do *GSM*)
EPC	*Electronic Product Code* (Código eletrônico do produto)
EUI-64	*Extended Unique Identifier 64 bits* (Identificador exclusivo estendido de 64 *bits*)
FM	*Frequency Modulation* (Modulação em Frequência)
FSK	*Frequency Shift Keying* (Modulação por chaveamento de frequência)
Gbps	*Giga bits* por segundo
GFSK	*Gaussian Frequency Shift Keying* (Modulação por chaveamento de frequência Gaussiana)
GHz	*Giga Hertz*
GPRS	*General Packet Radio Service* (Serviços Gerais de Pacote por Rádio)
GSM	*Global System for Mobile Communications* (Sistema Global para Comunicações Móveis)
HSPA	*High Speed Packet Access* (Acesso a pacotes de alta velocidade)
HTTP	*HyperText Transfer Protocol* (Protocolo de Transferência de Hipertexto)
I2C	*Inter-Integrated Circuit* (Circuito Inter-Integrado)
IBM	*International Business Machines* (Máquinas de Negócios Internacionais)
IDS	*Intrusion Detection Systems* (Sistemas de Detecção de Intrusão)
IEC	*International Electrotechnical Commision* (Comissão Eletrotécnica Internacional)
IED	*Intelligent Electronic Device* (Dispositivo Eletrônico Inteligente)
IEEE	*Institute of Electrical and Electronics Engineers* (Instituto de Engenheiros Elétricos e Eletrônicos)

IoT	*Internet of Things* (Internet das Coisas)
IPv6	*Internet Protocol version 6* (Protocolo de Internet versão 6)
ISM	*Industrial Scientific and Medical* (Industrial Científico e Médico)
Kb	kilo *byte*
Kbps	kilo *bits* por segundo
kHz	kilo *Hertz*
km	Quilômetros
LED	*Light Emissor Diode* (Diodo Emissor de Luz)
LiFi	*Light Fidelity* (Fidelidade a Luz)
LoRaWAN	*Long Range Wide Area Network* (Redes de longo alcance)
LPWAN	*Low Power Wide Area Network* (Redes de longo alcance e baixo consumo de energia)
LTE	*Long Term Evolution* (Evolução a Longo Prazo)
M	Metros
mA	Mili Ampere
MAC	*Media Access Control* (Controle de acesso de mídia)
Mbps	*Mega bits* por segundo
MHz	*Mega Hertz*
mm	milímetro
MQTT	*Message Queuing Telemetry Transport* (Transporte de Telemetria do Serviço de Enfileiramento de Mensagens)
ms	Milésimo de segundo
NBR	Normas Brasileiras de Regulação
NFC	*Near Field Communication* (Comunicação de campo próximo)
NTP	*Network Time Protocol* (Protocolo de tempo para redes)
ºC	Grau Celsius
OLED	*Organic Light-Emitting Diode* (Diodo emissor de luz orgânico)
OOK	*On-Off Keying* (Chaveamento através de liga desliga)
QoS	*Quality of Service* (Qualidade de serviço)
RAM	*Random Access Memory* (Memória de acesso aleatório)
REI	Redes Elétricas Inteligentes
RF	Rádio Frequência
RFID	*Radio-Frequency IDentification* (Identificação de rádio frequência)

RFID-HF	*Radio-Frequency IDentification-High Frequency* (Identificação de rádio frequência usando altas frequências)
RFID-LF	*Radio-Frequency IDentification-Low Frequency* (Identificação de rádio frequência usando baixas frequências)
RFID-UHF	*Radio-Frequency IDentification-Ultra High Frequency* (Identificação de rádio frequência em frequências extremamente altas)
RS-232	*Recommended Standard 232* (Padrão recomendado 232)
RS-485	*Recommended Standard 485* (Padrão recomendado 485)
RSSI	*Received Signal Strength Indication* (Indicação de força do sinal recebido)
RTU	*Remote Terminal Unit* (Unidade terminal remota)
RX	Recepção
SCADA	*Supervisory Control and Data Acquisition* (Controle de Supervisão e Aquisição de Dados)
SF	*Spreading Factor* (Fator de espalhamento)
SG	*Smart Grid* (Rede inteligente)
SHA	*Secure Hash Algorithm* (Algoritmo de hash seguro)
SPI	*Serial Peripheral Interface* (Interface periférica serial)
TCP	*Transmission Control Protocol* (Protocolo de Controle de Transmissão)
THz	*Tera Hertz*
TTL	*Transistor-Transistor Logic* (Lógica Transistor-Transistor)
TX	Transmissão
UDP	*User Datagram Protocol* (Protocolo de datagrama do usuário)
UMTS	*Universal Mobile Telecommunications System* (Sistema universal de telecomunicações móveis)
USB	*Universal Serial Bus* (barramento serial universal)
V	*Volt* (Tensão)
Vdc	Tensão Contínua
VHF	*Very High Frequency* (Frequência muito alta)
W	*Watt*
WiFi	*Wireless Fidelity* (Fidelidade sem fio)

SUMÁRIO

1 INTRODUÇÃO

Com o passar dos tempos, percebe-se a necessidade de obtermos cada vez mais informações em diversos segmentos da sociedade moderna e isso não é diferente na área da tecnologia, onde observamos o surgimento de cada vez mais tecnologias que permitem receber dados em qualquer lugar.

Uma das áreas em que se verifica grande avanço na aquisição de informações constantemente é a de energia elétrica, a qual é utilizada em todos os setores. Hoje é de extrema importância observar sistemas de geração, distribuição e consumo de energia e, com essas necessidades, surgiram as Redes Elétricas Inteligentes (REI), também conhecidas como *Smart Grid* (SG) (termo proveniente do inglês).

Com o advento das redes elétricas inteligentes, todas as etapas podem ser gerenciadas, tornando o sistema elétrico eficiente e mais confiável.

Atualmente já existem inúmeros dispositivos de medição operando em residências, comércios e indústrias, cujas informações são lidas remotamente e, em alguns modelos, é possível a realização de desligamentos por falta de pagamento, bem como, em sistemas de transmissão e distribuição, é possível a realização de chaveamentos através de chaves seccionadoras remotas, procedimento que antigamente era realizado manualmente.

Outros aspectos também são pertinentes as Redes Elétricas Inteligentes, como energias renováveis e veículos elétricos.

Neste trabalho, serão abordadas questões relacionadas aos meios e protocolos de comunicação, assim como questões de segurança dos dados que por estes são transportados.

1.1 Tema

Utilização de redes LPWAN (*Low Power Wide Area Network*) para gerenciamento de dispositivos *smart grid*.

1.2 Delimitação do Tema

O tema deste trabalho é a realização da comunicação de um dispositivo *smart grid*, que utiliza protocolo DNP3 e interface RS-232 (*Recommended Standard 232*),

desenvolvendo um conversor DNP3 para LoRaWAN, que transmite as informações para um *gateway* LoRaWAN previamente construído, disponibilizando assim, as informações para o servidor de rede que, por sua vez, disponibilizará as informações para o servidor de aplicação através do protocolo MQTT (*Message Queuing Telemetry Transport*).

1.3 Problema

O problema a ser resolvido é a medição e gerenciamento remoto de dispositivos medidores de energia, uma vez que, devido às grandes quantidades destes dispositivos em nossas residências, comércios e indústrias, torna-se inviável financeiramente a utilização de redes móveis provenientes da tecnologia celular.

1.4 Objetivos

Neste tópico, serão expostos os objetivos do presente trabalho, agora que o problema a ser avaliado foi devidamente apresentado.

1.4.1 Objetivo Geral

O objetivo geral trata do desenvolvimento de um sistema de comunicação para sistemas *smart grid* que se comuniquem através do protocolo DNP3 utilizando, como meio de comunicação, a tecnologia sem fio que opera nas faixas de frequência ISM (*Industrial Scientific and Medical*) 902 - 928 MHz (*Mega Hertz*), que são reservadas para desenvolvimento de sistemas industriais, científicos e médicos, sendo utilizada a modulação LoRa, que tem por objetivo a transmissão de dados a longas distâncias utilizando o protocolo de comunicação LoRaWAN.

1.4.2 Objetivos Específicos

Os objetivos específicos são:

a) validação em campo ponto a ponto, de um módulo LoRa específico, com relação ao alcance, taxa de transferência e consumo, de modo a caracterizar o módulo;

b) repetindo os testes acima, em uma rede utilizando *gateway*, com inúmeros dispositivos conectados, de modo a verificar questões de interferências e conflitos e colisões de pacotes;
c) realização da implementação do protocolo DNP3 trafegando sobre o meio de transmissão LoRa, utilizando o protocolo LoRaWAN;
d) avaliação de desempenho do protocolo DNP3 quando utilizado sobre o LoRaWAN, verificando se os requisitos da norma DNP3 são atendidos.

1.5 Justificativa

Considerando a grande quantidade de medidores e outros dispositivos que as concessionárias de energia utilizam em suas redes elétricas, torna-se necessária uma solução de baixo custo e longo alcance para realizar o gerenciamento remoto, dado que as soluções oriundas da telefonia móvel, como GSM, 3G, 4G, entre outras, tornam-se inviáveis devido ao custo mensal, além do problema de cobertura em locais mais distantes dos grandes centros e, por fim, tecnologias cabeadas como par trançado e fibra óptica, acabam por necessitar de grande infraestrutura física em seus acessos e, consequentemente, elevado custo de implantação, sendo necessário, em ambos casos, o uso de conversores RS-232, visto que os dispositivos *smart grid* normalmente dispõem desta interface.

2 FUNDAMENTAÇÃO TEÓRICA

Para a formulação deste trabalho, foi necessário explorar diversos tópicos que se relacionam direta e indiretamente com o tema, tornando assim o desenvolvimento do trabalho mais embasado em diversos aspectos.

2.1 Meios de Comunicação

Desde os primórdios, o homem tem a necessidade de se comunicar e, com o avanço das tecnologias a cada dia, surgem novas formas de nos comunicarmos.

Num passado não tão distante, achávamos extraordinário receber cartas. Com o passar do tempo, tivemos o advento das centrais telefônicas que, mesmo tendo a intervenção de outra pessoa “telefonista”, já nos permitiam conversar em tempo real com outras pessoas. E hoje, com o avanço das telecomunicações em praticamente qualquer parte do planeta, conseguimos realizar vídeo conferências.

Os meios de comunicação atualmente são divididos conforme o modo em que operam, podendo ser elencados como de comunicação guiada, onde é necessário um meio físico para transmissão e o de comunicação irradiada, onde ondas eletromagnéticas ou sinais de luz são irradiados diretamente. A seguir, podem ser vistos maiores detalhes com relação a estes meios.

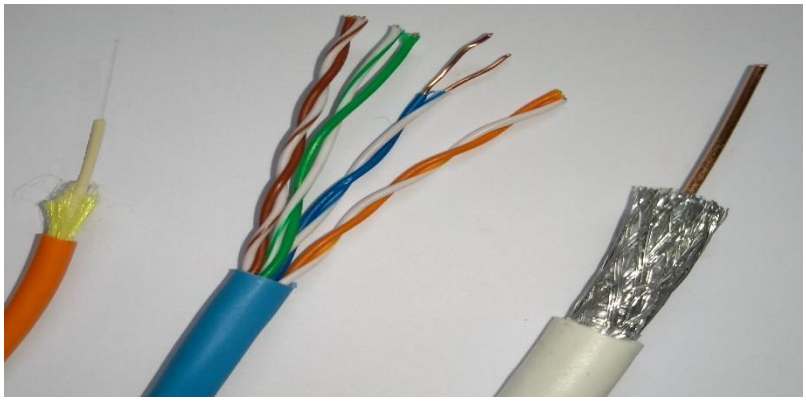
2.1.1 Comunicação guiada

A comunicação guiada é a utilização de meios físicos para a realização de transmissão e recepção de sinais elétricos ou ópticos, onde temos, como exemplos, o par trançado, o cabo coaxial e a fibra óptica. (TANENBAUM, 2003).

Existem muitas questões relacionadas a utilização desses meios de comunicação, que vão desde a taxa de transferência muito baixa, alcance limitado e elevado custo de construção. (TANENBAUM, 2003).

Na Figura 1, podem ser vistos alguns dos meios para comunicação guiada mais utilizados, como o cabo de fibra óptica, o cabo de par trançado e, por fim, o cabo coaxial. Estes meios de comunicação ainda são muito utilizados devido a sua confiabilidade e redes já implementadas pelas operadoras de comunicação no mundo.

Figura 1 – Exemplos de meios para comunicação guiada



Fonte: Elaborado pelo autor.

A seguir outro meio de comunicação, que não utiliza meio físico.

2.1.2 Comunicação irradiada

A comunicação irradiada é um tipo de comunicação que não necessita de meios físicos e se dispersa diretamente no ambiente. Em alguns casos, como na comunicação infravermelha e em alguns tipos de rádio, é necessário que os dispositivos estejam alinhados (visada) para conseguirem transmitir suas informações. Já em outros modos de rádio, existem vários modelos que não necessitam de visada para comunicar, pois acompanham a curvatura da terra, ficando restritas em algumas camadas da atmosfera como, por exemplo, o VHF (*Very High Frequency*). (TANENBAUM, 2003).

Com o avanço das tecnologias, novos meios surgiram e, com o passar dos anos, tornaram-se utilizadas diariamente, tais como satélites, GSM (*Global System for Mobile Communications*), WiFi (*Wireless Fidelity*), *Bluetooth* e rádios AM/FM (*Amplitude Modulation/Frequency Modulation*). (TANENBAUM, 2003).

Este também é um meio de comunicação utilizado nos dias atuais, principalmente em sistemas de redes celulares e de transmissão de dados e voz em sistemas de *backbone*, onde existe grande dificuldade para se chegar com algum meio guiado, como fibra-óptica ou par trançado. Convém ressaltar que seu custo de implantação pode ser elevado, quando comparado ao meio guiado, pois necessita de licenciamento em caso de rádios ou, então, de lançamento de satélites.

No próximo tópico será visto o conceito de informação, sendo este fundamental para a existência dos meios de comunicação.

2.2 Informação

O conceito de informação segundo Prado e Souza (2014, cap. 1) é,

> O conceito de informação envolve aspectos que abrangem desde sua coleta na forma bruta (dados), conduzindo ao processamento, que pode ser sob a forma de agrupamentos, cálculos, transformação dos dados, até sua disponibilização para a tomada de decisão. Dados são representações de fatos, podendo ser letras, números, imagens, sons, nomes etc., desprovidos de significados. A informação está vinculada à capacidade de relacioná-la ao contexto ao qual pertence, podendo estar associada a uma ação ou regra.

A informação é uma das coisas que mais necessitamos nos dias atuais e, portanto, é de extrema importância sabermos o que é fundamental para utilizarmos, de modo adequado, os meios de comunicação.

2.2.1 Relevância das informações

Com o aumento de dispositivos e sistemas que produzem informações, há a necessidade de transmissão e armazenamento delas. Portanto, torna-se essencial que estes dados sejam úteis para tomada de decisão, de modo que não gerem apenas volume e, sim, que sejam àqueles dados essenciais e necessários ao processo ao qual se destinam. Embora o custo de armazenamento de dados tenha tido redução com o passar dos anos devido a evolução de tecnologias, faz-se necessário o armazenamento daqueles que sejam úteis à tomada de decisão a qual servirão como base. (PRADO; SOUZA, 2014).

Ter em mente a relevância das informações é importante visto que, em muitos casos, estas podem ser armazenadas. Portanto, tratar a integração com sistemas pode evitar a duplicidade de dados ou, então, novas requisições utilizando mais banda dos meios de comunicação, conforme pode ser visto no tópico seguinte.

2.2.2 Integração

A integração é uma parte importante após a aquisição de informação, uma vez que, devido ao gerenciamento incorreto, pode-se produzir “ilhas de dados”, culminando uma repetição na obtenção de dados já existentes que, devido aos sistemas não serem integrados, produz-se informações duplicadas e retransmissões

desnecessárias, gerando utilização indevida dos meios de transmissão. (PRADO; SOUZA, 2014).

Com a integração de sistemas, vem outro fator importante, que é o de controle e segurança das informações, conforme pode ser visto no tópico a seguir.

2.2.3 Controle e segurança das informações

Conforme apresentado na norma ABNT (2005, p. x),

> A informação é um ativo que, como qualquer outro ativo importante, é essencial para os negócios de uma organização e consequentemente necessita ser adequadamente protegida. Isto é especialmente importante no ambiente dos negócios, cada vez mais interconectado. Como um resultado deste incrível aumento da interconectividade, a informação está agora exposta a um crescente número e a uma grande variedade de ameaças e vulnerabilidades [...].

A preocupação com o sigilo das informações vem bem antes do advento da *Internet* e computação. Logo, isso justifica que se faz necessário o extremo cuidado com transmissões, tratamento, armazenamento e compartilhamento de dados pois, em muitos casos, essas informações são cruciais para muitos processos dentro das organizações. (PRADO; SOUZA, 2014).

Existem inúmeras políticas de segurança, dentre as quais destacam-se a criptografia dos dados, que transforma o dado bruto em informação codificada, o *firewall*, que limita e controla o acesso e o IDS (*Intrusion Detection Systems*), que detecta intrusão. (PRADO; SOUZA, 2014).

Conforme visto neste tópico, a informação é um bem muito importante e, embora ela não possa ser tocada fisicamente, ela pode ser facilmente acessada caso as providências de segurança necessárias não sejam tomadas previamente. Isto porque, hoje em dia, tudo está sendo conectado a *Internet*, graças ao advento da *Internet* das Coisas, conforme será visto no capítulo seguinte.

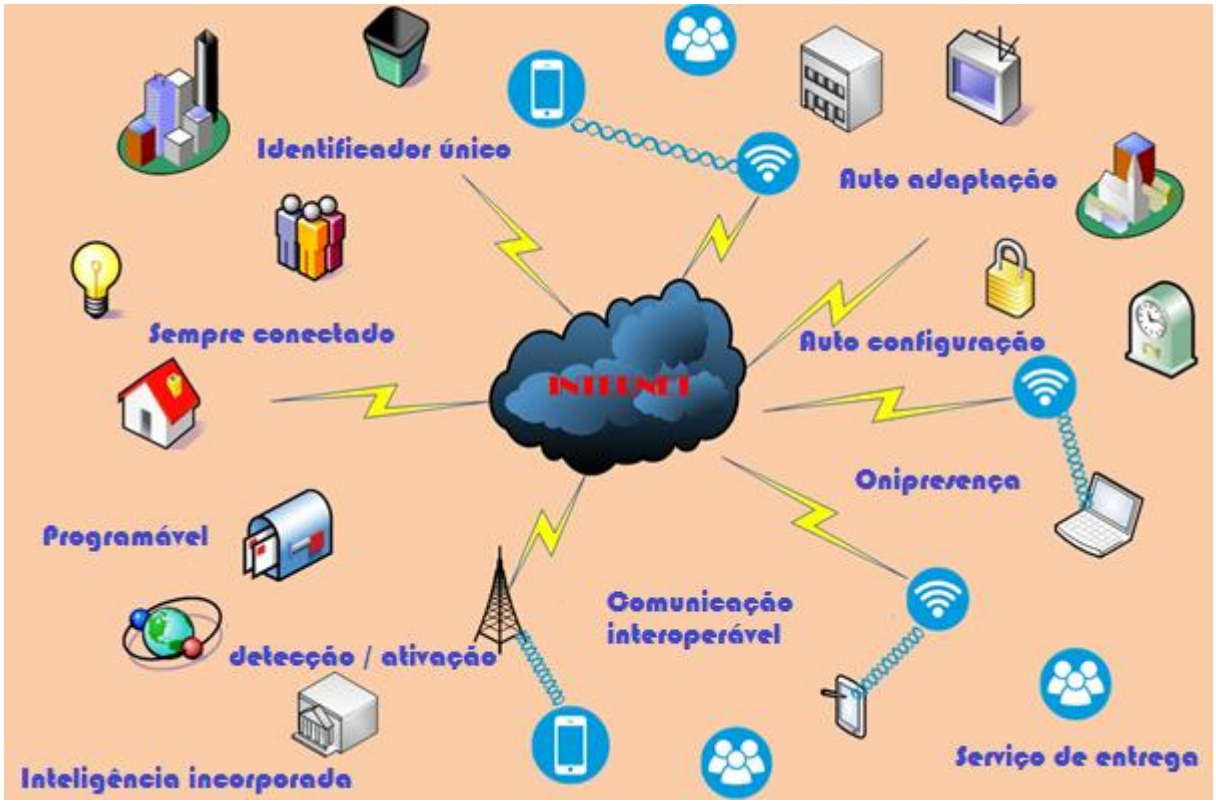
2.3 IoT *Internet* das Coisas

A definição de IoT (*Internet of Things*) é uma tarefa muito complexa, pois as utilizações desta terminologia são as mais variadas possíveis e não param de emergir novas. Porém, pode-se definir brevemente como uma rede complexa que se adapta facilmente aos dispositivos, sendo autoconfigurável, dado que trabalha com ID único

para cada dispositivo. Essa rede conecta “coisas” que, por sua vez, são dispositivos eletrônicos, tornando possível gerenciá-los remotamente, realizando acionamentos ou leituras de informações curtas. (MINERVA; BIRU; ROTONDI, 2015).

Na Figura 2, é possível ver inúmeras ilustrações de dispositivos que podem e estão sendo conectados a *Internet*, devido a necessidade cada vez maior de sabermos informações sobre processos ou atividades que acontecem em outros locais onde não estamos presentes fisicamente no momento.

Figura 2 – Características e escopo de um sistema IoT



Fonte: Adaptado de MINERVA, BIRU, ROTONDI (2015, p. 74).

Temos, como aspecto interessante, o surgimento de inúmeros protocolos de comunicação, os quais objetivam padronizar o modo que as informações são transmitidas entre os dispositivos “coisas”. Isso se deve muito à necessidade da utilização de pouca banda das redes, questões de segurança das informações que ali trafegam e diminuição do consumo de energia dos transmissores, uma vez que estes tendem a ser energizados com baterias para maior disponibilidade. (MINERVA; BIRU; ROTONDI, 2015).

Para melhor entendimento sobre o que é *Internet* das Coisas, é conveniente ver brevemente um pouco do histórico.

2.3.1 História

Na Segunda Guerra Mundial, radares eram utilizados para identificar a aproximação de aviões, mas com isso ainda não era possível identificar se eram aviões inimigos ou aliados. Então, os alemães descobriram que, ao realizar uma manobra de rotação antes de se aproximarem à sua base, alterava-se o sinal que o radar enviava à base, o qual era refletido nos aviões. Desse modo, foi possível diferenciar aviões inimigos e aliados. Esse sistema foi considerado o primeiro sistema RFID (*Radio-Frequency IDentification*) passivo. (MINERVA; BIRU; ROTONDI, 2015).

Anos depois, a tecnologia RFID continuou a ser explorada e foi muito utilizada em produtos vendidos em lojas, de modo que se tornaram etiquetas que tem por característica a identificação de um produto como pago ou não, tornando um sistema de combate ao furto, o que foi considerado como sistema RFID ativo. Outros usos para a tecnologia RFID foram implementados, desde a abertura de portas até a identificação de caminhões que transportam materiais nucleares. (MINERVA; BIRU; ROTONDI, 2015).

Em 1999 (mil novecentos e noventa e nove), um grupo de empresas e organizações se reuniu para financiar um projeto de pesquisa, com o objetivo de desenvolver etiquetas RFID de baixo custo, objetivando disponibilizá-las em todos os produtos de maneira a tornar possível a rastreabilidade destes apenas com a informação do número de série gravada. As informações seriam armazenadas em servidores de banco de dados acessíveis através da *Internet*. Com isto, o termo de *Internet* das Coisas foi utilizado pela primeira vez, em conjunto com o sistema de RFID/EPC (*Electronic Product Code*), sendo a base da conexão das "coisas" com a Internet. (MINERVA; BIRU; ROTONDI, 2015).

Conforme visto, a necessidade de termos informações sobre algo tem sido crescente desde a Segunda Guerra Mundial e hoje, cada vez mais dispositivos estão nos fornecendo informações em nosso dia a dia. A seguir, será vista a aplicação de *Internet* das Coisas.

2.3.2 Aplicações possíveis da IoT no dia a dia

Diversas são as aplicações possíveis de serem desenvolvidas com a *Internet* das Coisas. A aplicação denominada *Smart Home*, também conhecida como

Automação Residencial, é a que já temos visto tornar-se popular, com a qual é possível controlar e monitorar sistemas de ar condicionado (ventilação, aquecimento e refrigeração), iluminação, eletrodomésticos e sistemas de segurança. Com esses dispositivos conectados, muitas rotinas de nosso dia-a-dia são passíveis de ser automatizadas, além de ser possível a interação com os medidores de energia inteligentes (*Smart Grid*), que nos auxiliam para redução do consumo de energia. (XIAO, 2018).

Já na área da saúde, dispositivos que monitoram os sinais vitais de uma pessoa identificam quedas e/ou outros acidentes, acionando os serviços de emergência e/ou seus familiares, proporcionando um maior conforto e segurança para pessoas idosas e deficientes físicos, que possuem mais limitações, ou, ainda, podendo melhorar o sistema de saúde, permitindo tratar remotamente os pacientes com o auxílio da telemedicina e cirurgias à distância através da utilização ou auxílio de robôs. (XIAO, 2018).

No setor de transporte, carros conectados juntamente com cidades inteligentes podem auxiliar a evitar regiões com problemas de engarrafamentos, alagamentos ou acidentes, encontrar um local para estacionar ou evitar acidentes. Ações voltadas para o transporte público poderiam beneficiar os usuários, sendo estes informados sobre atrasos, problemas de lotação de passageiros, podendo as organizações responsáveis tomar ações de modo a diminuir custos de manutenção e gerenciamento das frotas. (XIAO, 2018).

A área ambiental também pode ser beneficiada com a *Internet* das Coisas, uma vez que sensores podem medir a qualidade do ar, água, condições do solo, monitorar queimadas, avalanches, deslizamentos de terra e animais selvagens, possibilitando proteger a fauna e a flora em áreas de preservação ambiental e estudar o comportamento das espécies sem grandes intervenções nos habitats. A agricultura inteligente possibilita o monitoramento e aplicação no solo, de irrigação, fertilizantes e agrotóxicos, com o intuito de preservar o meio ambiente e reduzir custos de produção para o agricultor, uma vez que estes seriam aplicados conforme a necessidade de cada cultura ou região. (XIAO, 2018).

Esses foram alguns exemplos de aplicações que estão diretamente ligadas e nas quais podemos ter acesso as informações que são geradas, nos influenciando na tomada de decisões. A seguir, será vista a influência na indústria a qual utilizamos indiretamente.

2.3.3 Aplicações possíveis da IoT na indústria

Na indústria, a *Internet* das Coisas ajuda a criar a Quarta Revolução Industrial, também conhecida como Indústria 4.0, tendo, por objetivo, criar fábricas inteligentes, nas quais as máquinas podem utilizar inteligência artificial, auto configurando e auto otimizando, sendo assim mais eficientes e produzindo produtos com qualidade superior. (XIAO, 2018).

Um referencial histórico a respeito das outras fases da indústria nos remete ao século XVIII, onde houve o advento da máquina a vapor, sendo considerada a Primeira Revolução Industrial, enquanto a Segunda ocorreu no início do século XIX, com o auxílio da energia elétrica na produção em massa de produtos.

Por fim, a Terceira Revolução Industrial ocorreu no final do século XIX, quando a automatização com robôs começou a ser mais utilizada nos processos fabris, graças a evolução da eletrônica e da informática. (XIAO, 2018).

No tópico a seguir serão apresentados alguns tipos de redes para a *Internet* das Coisas, realizando um comparativo entre as redes.

2.3.4 Tipos de redes para *Internet* das Coisas

Além das tradicionais redes de comunicação, tais como *Ethernet*, WiFi e *Bluetooth*, existem outras que podem ser utilizadas para comunicação de dispositivos da Internet das Coisas. (XIAO, 2018).

Conforme falado anteriormente, considerado como precursor da *Internet* das Coisas, temos o RFID, o qual realiza a leitura de etiquetas conectadas em objetos através de ondas eletromagnéticas de radiofrequência, operando em diversas frequências. Essas etiquetas podem ser passivas, ou seja, não possuem fontes de energia ou baterias, sendo energizadas pelas ondas eletromagnéticas provenientes dos leitores, limitando o alcance em aproximadamente 25 metros. Já as etiquetas ativas, que contêm sua fonte própria de energia, podem ser lidas a até 100 metros de distância. (XIAO, 2018).

Já o NFC (*Near Field Communication*) opera na frequência de 13,56 Mhz, sendo baseado em comunicação ponto a ponto, significando que o dispositivo com NFC integrado pode ser um leitor ou uma etiqueta e, desta forma, não podem estar distantes entre si mais do que 4 centímetros. Atualmente, muitos telefones celulares

(*smartphones*) já dispõem desta tecnologia, que nos permite realizar pagamentos utilizando o próprio aparelho ou, então, repassar informações de um aparelho para outro, tais como contatos ou imagens, bastando aproximar dois aparelhos com a tecnologia. (XIAO, 2018).

Existe ainda a evolução do *Bluetooth*, conhecido como BLE (*Bluetooth Low Energy*), o qual é um *Bluetooth* de baixo consumo de energia, operando na mesma faixa de frequência de 2,4 GHz, porém utiliza-se de um sistema de modulação mais simples que o *Bluetooth* convencional, pois permanece em modo de suspensão em momentos que não está transmitindo ou recebendo informações, gerando um consumo muito menor de energia. (XIAO, 2018).

Através da utilização da luz visível, temos o LiFi (*Light Fidelity*), que é uma tecnologia nova, a qual não utiliza fios ou ondas eletromagnéticas e, sim, lâmpadas LED (*Light Emissor Diode*), cujo diferencial é já disporem de diodos emissores de luz, os quais, ao variarem a corrente elétrica muito rapidamente, tornam-se transmissores de informações que não são captadas pelos nossos olhos, somente pelos fotorreceptores, tendo como desvantagem o fato de não atravessarem paredes. Dado que essa tecnologia vem sendo cada vez mais utilizada no dia a dia, podem ser uma boa opção em residências e edifícios em geral. (XIAO, 2018).

Com o objetivo de utilizar o protocolo IPv6 (*Internet Protocol version 6*) e todos os protocolos de comunicação como o TCP (*Transmission Control Protocol*), UDP (*User Datagram Protocol*), HTTP (*HyperText Transfer Protocol*), MQTT, entre outros, temos a tecnologia 6LoWPAN (*IPv6 over Low Power Wireless Personal Area Networks*), que também visa o baixo consumo de energia e, consequentemente, baixa taxa de transferência, baseada na norma IEEE 802.15.4, possibilitando boa integração com a *Internet*, permitindo endereçar cada dispositivo com endereços IPv6.

Ainda, utilizando a mesma norma IEEE 802.15.4, temos o *ZigBee*, que se diferencia do 6LoWPAN por não dispor do suporte ao IPv6, porém com alcance maior, necessitando ter visada (sem barreiras) entre os receptores e os transmissores e com o custo menor que o 6LoWPAN. (XIAO, 2018).

Outra tecnologia com o objetivo de fornecer transmissões de baixa latência e com confiabilidade utilizando frequências abertas (ISM) é a *Z-Wave*, a qual tem um baixo alcance, aproximadamente 100 metros e uma taxa de transmissão relativamente baixa, cuja aplicação maior é em automação residencial. (XIAO, 2018).

Por fim, temos a tecnologia LoRa, que é uma tecnologia para comunicação de longo alcance na faixa dos quilômetros de distância, sendo energizada por baterias devido ao seu baixo consumo de energia (LPWAN), pois opera com baixa potência de transmissão na faixa de frequência (ISM), suportando comunicações bidirecionais em baixíssimas taxas de transmissão. (XIAO, 2018).

No Quadro 1, são apresentadas informações de alcance e taxa de transferência sobre as redes de comunicação utilizadas em IoT, para conexão dos dispositivos.

Quadro 1 – Comparativo das redes para Internet das Coisas

Tecnologia	Frequência de Operação	Alcance	Taxa de dados
RFID-LF (passivo)	125–134,2 kHz	10 m	-
RFID-HF (passivo)	13,56 MHz	10 cm - 1 m	-
RFID-UHF (passivo ou ativo)	433 MHz	1–100 m	-
	856–960 MHz	1–12 m	-
RFID *Microwave* (ativo)	2,45–5,8 GHz	1–2 m	-
	3,1–10 GHz	<200 m	-
NFC	13,56 MHz	10 cm	100–420 kbps
LiFi	400–800 THz	<10 m	<224 Gbps
WiFi	2,4 GHz e 5 GHz	~50m	<1 Gbps
GSM/GPRS/EDGE(2G), UMTS/HSPA(3G), LTE (4G), 5G	900, 1800, 1900 e 2100 MHz 2,3; 2,6; 5,25; 26,4 e 58,68 GHz	<200 km	<500 kbps (2G), <2 Mbps (3G), <10 Mbps (4G) <100 Mbps (5G)
Bluetooth (BLE) (4.2)	2,4 GHz	50–150 m	1 Mbps
6LoWPAN	2,4 GHz ~1 GHz	<20 m	20–250 kbps
ZigBee	2,4 GHz	10–100 m	250 kbps
Z-Wave	868,42 MHz e 908,42 MHz	100 m	<100 kbps
LoRa	868 MHz e 915MHz	<15 km	0,3–50 kbps

Fonte: Xiao (2018, p.32 e 35).

No item seguinte, visto que o objetivo deste trabalho são soluções que envolvam redes sem fio que utilizam LoRa, serão apresentados os tipos de modulações de rádio.

2.4 Modulações de rádio

Este tópico tem por objetivo apresentar algumas modulações de rádio muito utilizadas em diversos dispositivos de comunicação, assim como a modulação LoRa, que é utilizada como base nos módulos utilizados neste trabalho.

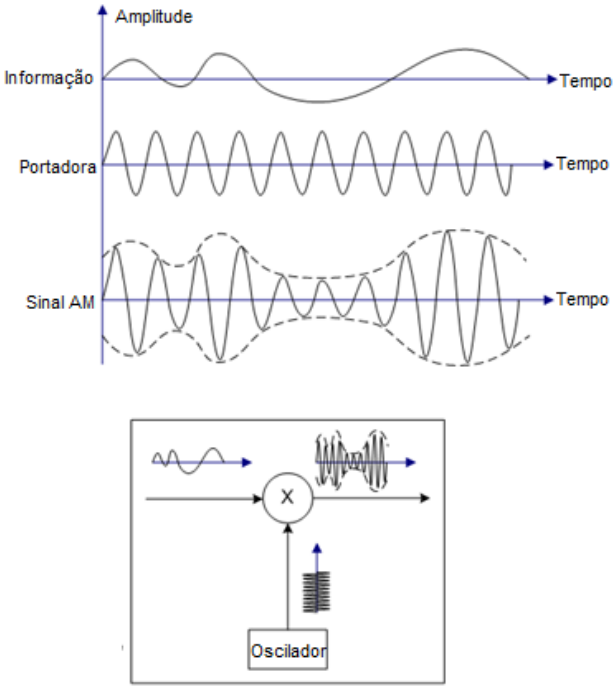
2.4.1 Modulação em Amplitude (AM)

Muito utilizada no nosso dia a dia em emissoras de rádio e radioamadores, conhecida popularmente por AM.

Essa modulação, por sua vez, opera variando a amplitude (intensidade) do sinal na onda portadora proporcionalmente ao sinal a ser transmitido. Cabe considerar que essa modulação é mais suscetível a ruídos, causando problemas indesejados, porém, em contrapartida, tem grande alcance de transmissão. (SENEVIRATNE, 2019).

Na Figura 3, é possível verificar os dois sinais necessários para a formação da modulação em amplitude, sendo estes a informação a ser transmitida e a portadora, que após serem misturadas em “X”, que é denominado de *mixer*, temos como resultado o “Sinal AM”, que é o sinal em modulação de amplitude. (SENEVIRATNE, 2019).

Figura 3 – Sinal de Amplitude Modulada



Fonte: Adaptado de SENEVIRATNE (2019, p. 3).

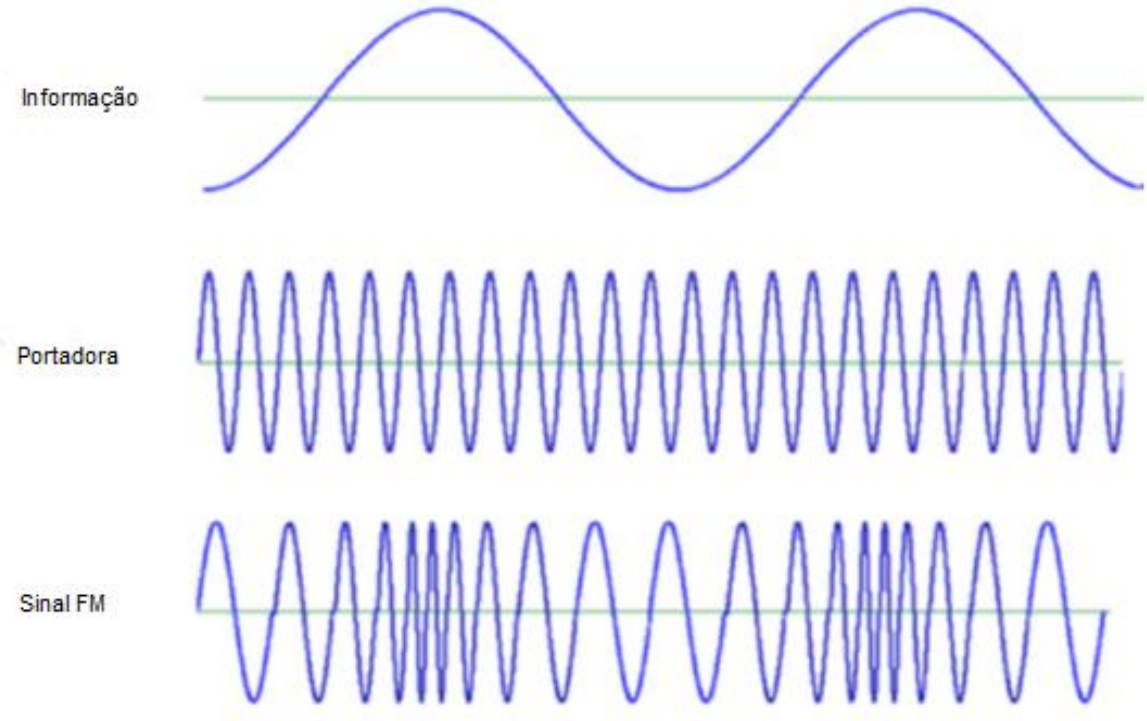
2.4.2 Modulação em Frequência (FM)

Assim como visto na modulação AM, a modulação em frequência, também conhecida por FM, é muito utilizada em emissoras de rádio, sendo também de grande utilização no cotidiano.

Essa modulação, por sua vez, opera variando a frequência do sinal na onda portadora de acordo com a amplitude do sinal. Cabe considerar que essa modulação é menos suscetível a ruídos. Em contrapartida, seu alcance de transmissão é bem menor quando comparado ao AM. (SENEVIRATNE, 2019).

Na Figura 4, é possível verificar os dois sinais necessários para a formação da modulação em frequência, sendo estes a informação a ser transmitida e a portadora, que após serem misturadas em um *mixer*, temos como resultado o “Sinal FM”, que é o sinal em modulação de frequência. (SENEVIRATNE, 2019).

Figura 4 – Sinal de Frequência Modulada



Fonte: Adaptado de SENEVIRATNE (2019, p. 4).

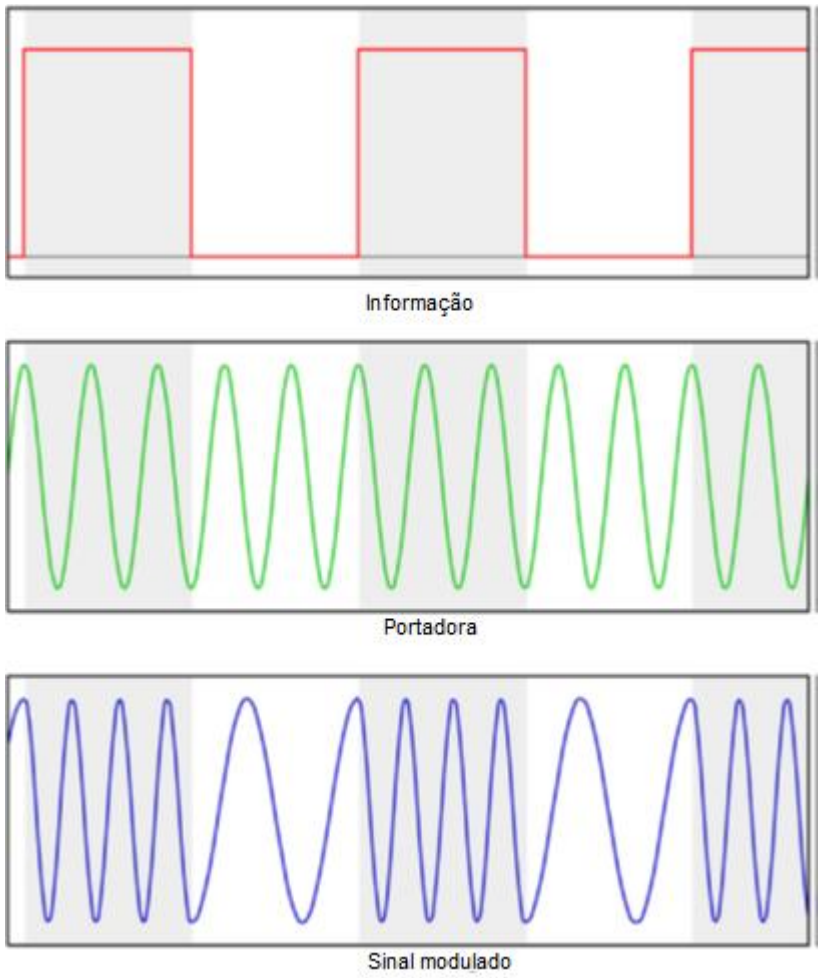
2.4.3 Modulação: *Frequency Shift Keying* (FSK)

Conforme visto anteriormente na modulação FM, a modulação FSK também se baseia na alteração da frequência. Essa modulação é bastante utilizada em sistemas de alarmes, ou seja, em sistemas de baixa potência.

Nesta modulação são utilizadas duas frequências diferentes, uma representando o sinal digital 1 e a outra o sinal digital 0, ou também, chamados de estado alto e baixo, respectivamente. (SENEVIRATNE, 2019).

Na Figura 5, é possível verificar os dois sinais necessários para a formação da modulação em amplitude, sendo estes, a informação a ser transmitida e a portadora, após serem misturadas em um *mixer*, temos como resultado o “Sinal Modulado”, que é o sinal analógico modulado em frequência. (SENEVIRATNE, 2019).

Figura 5 – Sinal da modulação FSK



Fonte: Adaptado de SENEVIRATNE (2019, p. 5).

2.4.4 Modulação: *Chirp Spread Spectrum* (CSS)

Nas modulações CSS e FSK, uma das características que mais prevalece é a operação em baixas potências. Assim, a modulação CSS que, inicialmente, em 1940, foi desenvolvida para aplicações em sistemas de radares, com o passar do tempo, devido as suas características de baixa potência, longo alcance e menos suscetibilidade a interferências, passou a ser utilizada no ramo de comunicações nas áreas militares e espaciais. (SENEVIRATNE, 2019).

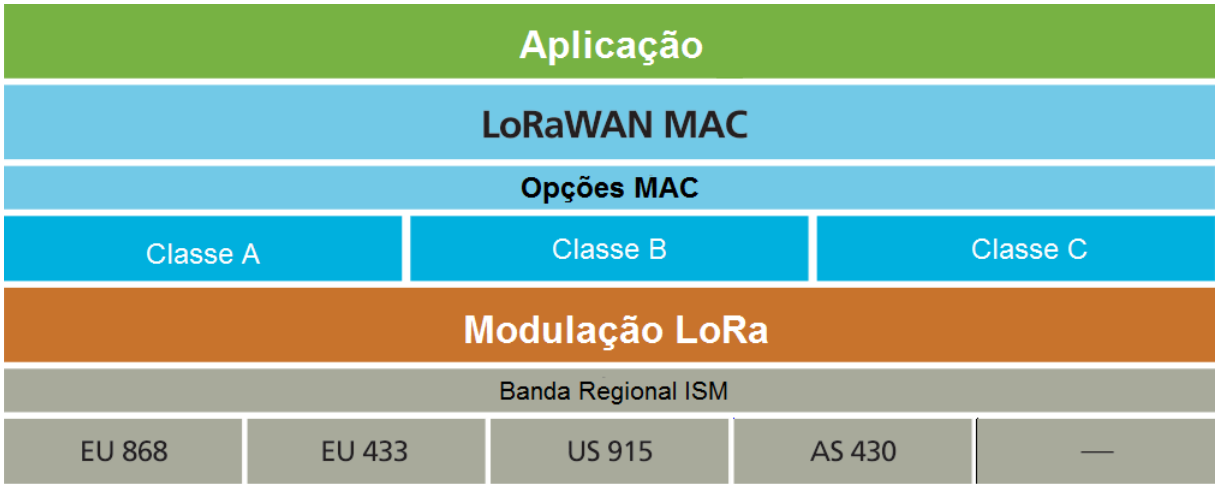
Seu modo de funcionamento consiste no espalhamento espectral de pulsos *CHIRP*, que são modulados em frequência com o objetivo de codificar as informações que por ali trafegam. (SENEVIRATNE, 2019).

2.4.5 Modulação: *LoRa Spread Spectrum*

A modulação LoRa, derivada da modulação CSS, vista no tópico anterior, sendo desenvolvida pela empresa americana *Semtech*, tem o objetivo de interconectar dispositivos a IoT.

Na Figura 6, é possível ver todas as camadas que compõem as partes físicas e lógicas. Conforme visto no item “Tipos de redes para IoT”, usa-se a faixa de frequência ISM, que não necessita ser licenciada. (SEMTECH, 2019).

Figura 6 – Camadas físicas e lógicas do LoRa



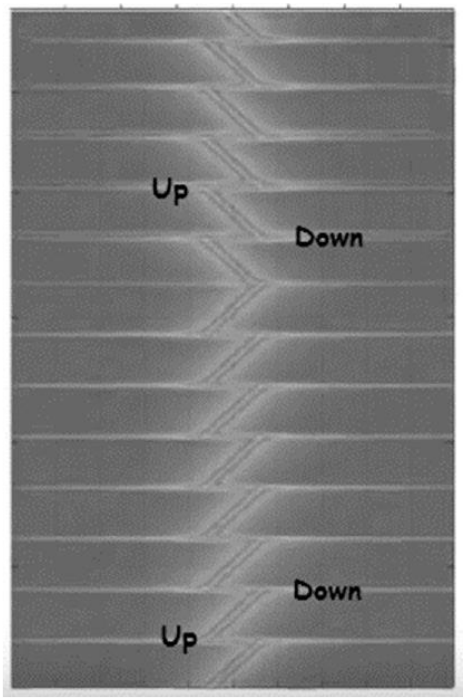
Fonte: Adaptado de SEMTECH (2019).

Essa modulação usa o espectro de propagação *CHIRP* para codificar os dados, sendo que cada *bit* (*Binary digit*) é espalhado por um fator de espalhamento denominado SF (*Spreading Factor*). (SENEVIRATNE, 2019).

Os fatores de espalhamento SF utilizados nessa modulação variam de 7 a 12, quanto menor o fator de espalhamento, maior a taxa de transferência, ficando um tempo menor transmitindo. Já em fatores de espalhamento maiores, menor será a taxa de transmissão, ficando um tempo maior transmitindo, o que possibilita maior alcance de transmissão. (SENEVIRATNE, 2019).

Como já visto na modulação CSS, a modulação LoRa é mais complexa e mais resistente a interferências, uma vez que utiliza varredura entre duas frequências, conforme pode ser visto na Figura 7, sendo que a parte superior da Figura 7 mostra a varredura de frequência de baixo para cima e a parte inferior da Figura 7 mostra a varredura de frequência de cima para baixo. (SENEVIRATNE, 2019).

Figura 7 – Varredura de frequências da modulação LoRA



Fonte: SENEVIRATNE (2019, p. 7).

Até o presente momento, foram vistos os meios de comunicação e tipos de modulações de rádio existentes, sendo que isto está diretamente relacionado ao hardware.

No tópico seguinte, serão apresentados os protocolos mais utilizados para dispositivos de *Internet* das Coisas

2.5 Protocolos

Na área de automação, seja ela residencial, comercial ou industrial, sempre existe a necessidade de interagirmos com os dispositivos. Desta necessidade surgiram os protocolos de comunicação, que estabelecem regras de modo a padronizar a maneira com que as informações serão acessadas.

2.5.1 *Modbus*

O protocolo *Modbus* foi desenvolvido pela *Gould Modicon*, sendo agora *Schneider Eletric*, com o objetivo de servir a sistemas de controle de processo. *Modbus* é definido como um protocolo público, muito utilizado em equipamentos de automação de diversos fabricantes. (MAKAY; WRIGHT; REYNDERS; PARK, 2004).

Este protocolo não está ligado diretamente a interfaces físicas, uma vez que funciona desde interfaces seriais (RS232 e RS485) até interfaces *ethernet* (TCP/IP), visto que ele é extremamente simples, flexível e possui muitas publicações e documentações disponíveis na *Internet*. Outras características importantes do mesmo são o formato e a sequência dos comandos de dados, comunicação mestre (*master*) para escravo (*slave*), resposta a comandos malformados e modos de transmissão de dados RTU (*Remote Terminal Unit*) e ASCII (*American Standard Code for Information Interchange*). (MAKAY; WRIGHT; REYNDERS; PARK, 2004).

O Quadro 2 e o Quadro 3 apresentam exemplos dos formatos de quadros das mensagens de leitura e resposta, mestre-escravo no protocolo *Modbus*.

Quadro 2 – Exemplo comando de leitura – status saídas digitais

Endereço Escravo	Função	Registrador Inicial		Quantidade Registradores		CRC16	
0x01	0x01	0x00	0x0a	0x00	0x02	0x9d	0xc9

Fonte: MAKAY, WRIGHT, REYNDERS, PARK (2004, p.99).

Quadro 3 – Exemplo comando de resposta – status saídas digitais

Endereço Escravo	Função	Contador de Bytes	Dados da Resposta	CRC16	
0x01	0x01	0x01	0x03	0x11	0x89

Fonte: MAKAY, WRIGHT, REYNDERS, PARK (2004, p.99).

Este protocolo não contem criptografia ou qualquer outro método de segurança, exceto a validação do CRC (*Cyclic Redundancy Check*), nas mensagens que transitam, sendo assim facilmente interceptável e pouco seguro, mas serviu de base para o desenvolvimento do protocolo DNP3, que será apresentado no tópico a seguir.

2.5.2 DNP3

O protocolo DNP3 define um padrão de comunicação entre estações mestres e escravas, entre outros possíveis dispositivos inteligentes, tendo sido desenvolvido para auxiliar a interoperabilidade de sistemas nos mais diversos segmentos, como eletricidade, petróleo/gás, água/efluentes e segurança. (CLARKE; REYNDERS; WRIGHT, 2004).

Originalmente, o protocolo foi desenvolvido pela *Harris Controls Division*, para uso em seus equipamentos especificamente para a indústria de energia elétrica, sendo disponibilizado, em 1993, para uso em outras empresas, onde foi criado o Grupo de Usuários DNP3. Mediante o pagamento de uma taxa nominal, qualquer pessoa ou empresa pode ter acesso a especificação completa do protocolo, diferentemente de outros protocolos que tem suas especificações abertas na *Internet*. (CLARKE; REYNDERS; WRIGHT, 2004).

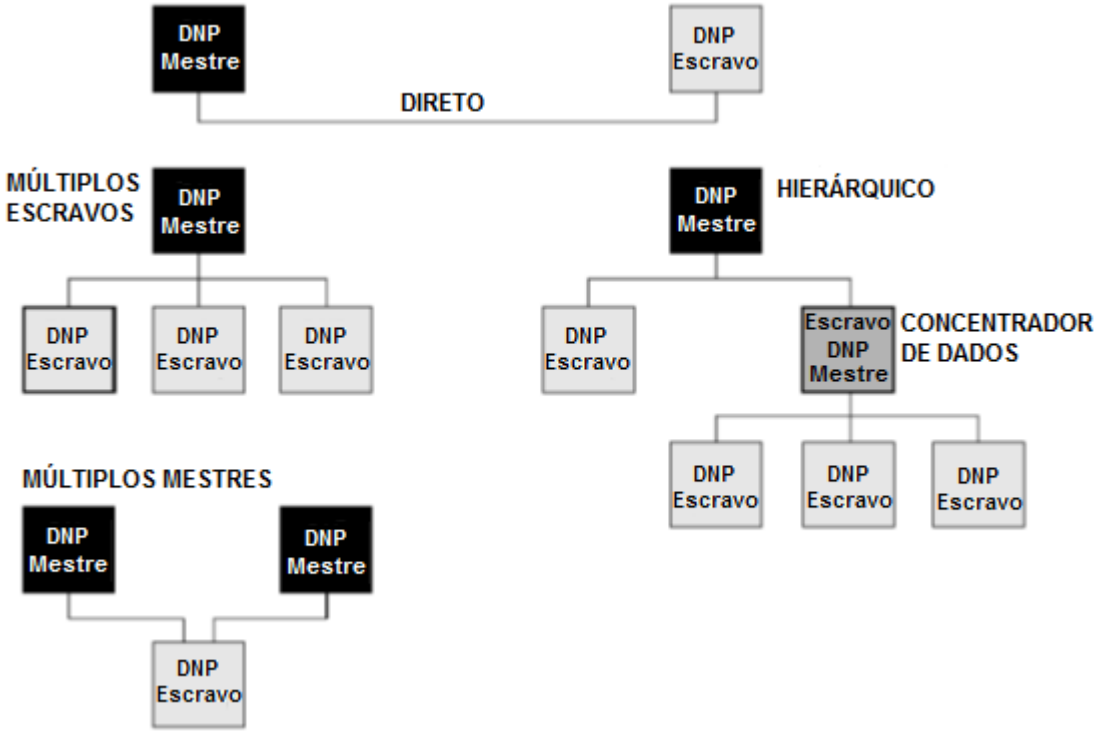
Desenvolvido para aplicações SCADA (*Supervisory Control and Data Acquisition*) de supervisão e aquisição de dados através de comandos entre um computador e os dispositivos que podem estar fisicamente distantes, foi projetado para envio de pacotes pequenos de dados de modo confiável e em forma sequencial, se diferenciando, assim, dos demais protocolos. (CLARKE; REYNDERS; WRIGHT, 2004).

Este protocolo é utilizado largamente em diversos setores na América do Norte, América do Sul, África do Sul, Ásia e Austrália. Na Europa, o protocolo IEC 60870-5-101 é o mais utilizado em sistemas de energia, enquanto em outros setores é utilizado o DNP3. É importante observar que o protocolo IEC 60870-5-101, compartilha sua origem com o protocolo DNP3. (CLARKE; REYNDERS; WRIGHT, 2004).

Neste trabalho será utilizada a topologia direta (*DIRECT*) ou (*One on One*), onde se tem um mestre (*master*) trocando informações com um escravo (*slave ou outstation*) diretamente sem intermediários, utilizando, como meio de transporte, os rádios LoRa, assim como o protocolo LoRaWAN, que será visto mais adiante.

Outros tipos de topologia podem ser visualizados na Figura 8:

Figura 8 – Topologias protocolo DNP3



Fonte: Adaptado de CLARKE; REYNDERS; WRIGHT (2004, p. 70).

Um aspecto interessante do protocolo DNP3 é a geração de eventos automaticamente pelo escravo, onde o mestre não necessita ficar consultando, constantemente, o equipamento escravo para verificar se houve alterações nas informações deste. Uma vez que ocorra alterações, estas serão automaticamente informadas ao mestre, tornando assim a utilização dos meios de comunicação mais otimizada. (CLARKE; REYNDERS; WRIGHT, 2004).

Outras características do protocolo DNP3 são: possibilidade de endereçar mais de 65000 dispositivos, mensagens com registro de data e hora para identificação sequencial, quebra de mensagens em diversos pacotes, controle de erros, comunicação segura e transferência de arquivos. (CLARKE; REYNDERS; WRIGHT, 2004).

Um fator muito importante no desenvolvimento do protocolo DNP3 é o nível de implementação que se tem, como objetivo, atender. Com isso, é possível definir o nível de interoperabilidade entre equipamentos mestre e escravo. Portanto, o nível DNP3-L1 é fundamental para garantir a interoperabilidade entre fabricantes. A seguir,

são apresentados os três níveis e suas características. (CLARKE; REYNDERS; WRIGHT, 2004).

DNP3-L1: Sendo a mais simples de realizar a implementação, destinando-se a comunicação de dispositivo mestre (*master*) ou intermediário (*data concentrator*) com um dispositivo IED (*Intelligent Electronic Device*) pequeno, sendo que pode ser um medidor, relé, controlador ou religador automático, disponibilizando assim, todas as entradas e saídas desses dispositivos, que normalmente são locais. (CLARKE; REYNDERS; WRIGHT, 2004).

DNP3-L2: Define um subconjunto maior de características do DNP para o DNP3-L1, destinando-se a comunicação de dispositivo mestre (*master*) ou intermediário (*data concentrator*) com um dispositivo IED grande ou uma RTU, disponibilizando assim, todas as entradas e saídas desses dispositivos, que normalmente são locais. (CLARKE; REYNDERS; WRIGHT, 2004).

DNP3-L3: Define um subconjunto maior de características do DNP, mas ainda não requer suporte a todos recursos possíveis do DNP, cobrindo os mais necessários e utilizados com maior frequência, destinando-se a comunicação de dispositivo mestre (*master*) ou intermediário (*data concentrator*) com um dispositivo RTU maior ou mais avançado, disponibilizando assim, todas as entradas e saídas desses dispositivos, que normalmente são locais e remotas. (CLARKE; REYNDERS; WRIGHT, 2004).

Até então, vimos exemplos de protocolos *Modbus* e DNP3, que foram desenvolvidos especificadamente para trabalhar com os equipamentos finais, que exercem automações diretamente na atividade ou no processo fim a que se destinam, enquanto os protocolos que MQTT e LoRaWAN, que serão vistos no próximo tópico, são desenvolvidos para trabalhar em conjunto com o meio de transmissão e publicação dos dados, respectivamente.

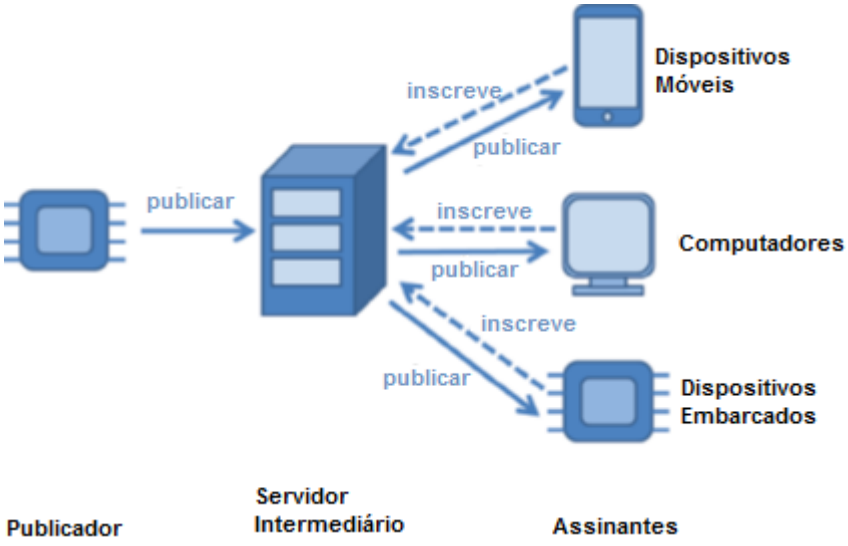
2.5.3 MQTT (*Message Queuing Telemetry Transport*)

Projetado pela IBM (*International Business Machines*) para uso em dispositivos voltados para a *Internet* das Coisas (IoT), o protocolo MQTT é extremamente leve, baseando-se no modelo publicador-assinante em inglês (*publisher-subscriber*), onde o publicador divulga, em um servidor intermediário, chamado *broker*, os dados de sensores e/ou variáveis de um sistema. Esse servidor intermediário é responsável pela publicação das informações, permitindo que assinantes que podem ser

dispositivos móveis, computadores ou outros dispositivos embarcados, recebam essas informações. (XIAO, 2018).

Na Figura 9, pode-se observar o funcionamento do protocolo MQTT, com todos os dispositivos envolvidos em uma publicação de dados.

Figura 9 – Exemplo estrutura de comunicação do protocolo MQTT



Fonte: Adaptado de XIAO (2018, p. 37).

Outros aspectos que demonstram sua utilização em sistemas IoT são baixa utilização de banda em transferência de dados, quantidade de dados transferidos relativamente pequenos, menor consumo de energia, o que facilita sua utilização em sistemas alimentados com baterias e a priorização de mensagens, com a utilização de QoS (*Quality of Service*), conforme demonstra o Quadro 4. (XIAO, 2018).

Quadro 4 – Tipos de QoS no protocolo MQTT.

QoS	**Cliente /Servidor**
0	entrega a informação uma vez, sem a necessidade de confirmação.
1	entrega a informação pelo menos uma vez, confirmação necessária.
2	entrega a informação apenas uma vez, usando negociação.

Fonte: XIAO (2018, p. 38).

Conforme visto no protocolo MQTT, um dispositivo publica em um servidor as informações que poderão ser acessadas por outros dispositivos como *smartphones*, computadores ou outros dispositivos que, por sua vez, também poderão publicar informações. Neste protocolo podemos cair no caso já levantado por este trabalho,

onde teremos a mesma informação em diversos lugares e, portanto, deve-se utilizar parcimônia. No tópico seguinte, será apresentado o protocolo LoRaWAN.

2.5.4 LoRaWAN (*Long Range Wide Area Network*)

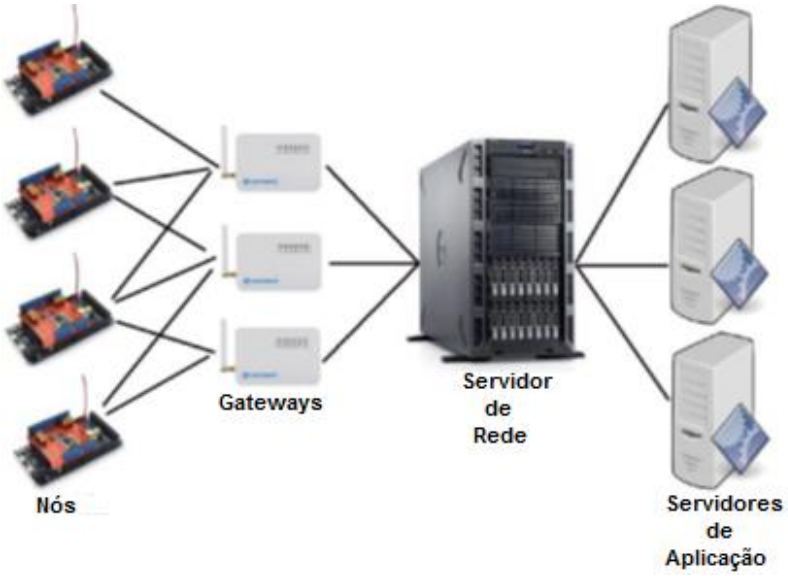
LoRaWAN é um protocolo de comunicação que arquiteta todo o sistema de comunicação, enquanto o LoRa estrutura a camada física do rádio que é responsável pelas transmissões de longo alcance, conforme já visto anteriormente.

Quanto ao protocolo, este é responsável, desde a transmissão dos dispositivos finais ou também conhecidos como “nós” (*end nodes*), até os dispositivos chamados de *gateways* ou concentradores, que acabam por centralizar os inúmeros “nós”, disponíveis em uma rede LoRa. (SENEVIRATNE, 2019).

Os *gateways* operam em topologia estrela, onde são responsáveis por divulgar as informações para servidores de rede (*Network Server*). Essa transmissão se dará por meios convencionais de comunicação, tais como redes celulares, *ethernet*, entre outras. Já os servidores de rede (*Network Server*), repassam essas informações para servidores de aplicação (*Application Servers*), disponibilizando assim, as informações para que sejam acessadas conforme necessidade e modo desenvolvidos. (SENEVIRATNE, 2019).

Na Figura 10, pode-se observar todos os dispositivos necessários para o funcionamento de uma rede que utiliza protocolo LoRaWAN.

Figura 10 – Topologia comunicação LoRaWAN

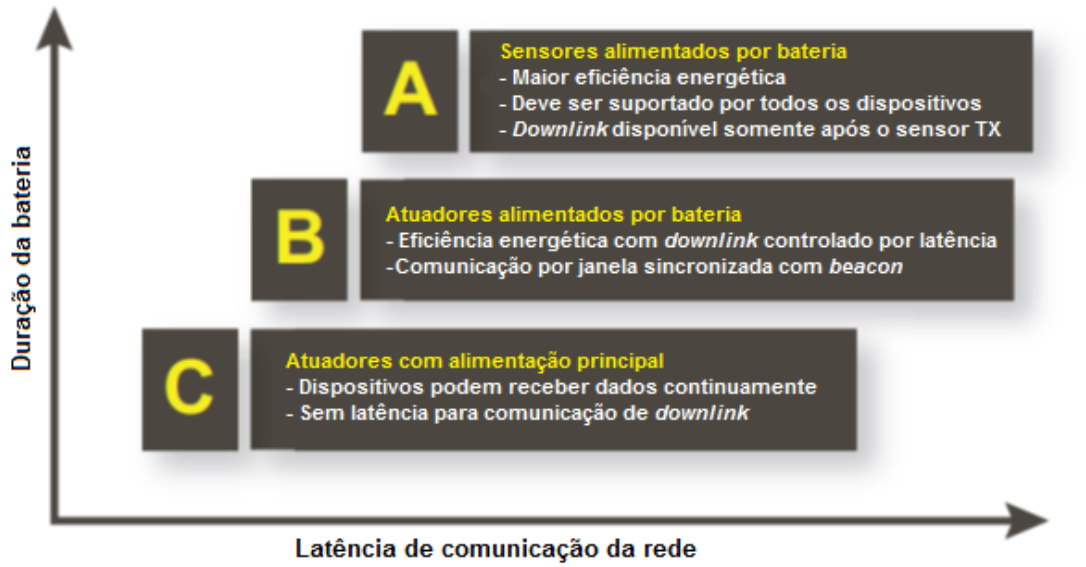


Fonte: Adaptado de SENEVIRATNE (2019, p. 13).

Outra característica importante para o protocolo LoRaWAN é questão do consumo de energia. Por este motivo, é adotado o modo de comunicação assíncrono, comunicando-se apenas quando houver dados para serem enviados (eventos) ou, então, por tempo (agendamento). Este método é conhecido por *ALOHA*, onde os “nós” sincronizam-se em intervalos de tempo. Deste modo, o consumo de energia é reduzido, enquanto em redes convencionais síncronas, frequentemente são verificadas se existem novas informações, consumindo mais energia. (LORA ALLIANCE, 2019).

O LoRaWAN dispõe de três classes de operação, de modo a atender as necessidades de cada aplicação, conforme os requisitos necessários, vide Figura 11.

Figura 11 – Classes de operação LoRaWAN



Fonte: Adaptado de LORA ALLIANCE, (2019).

Classe A: Opera de modo bidirecional com potência reduzida, devendo ser suportada por todos os dispositivos finais, uma vez que a comunicação é totalmente assíncrona, sempre iniciando neste, ou seja, o dispositivo final envia para o *gateway* informações logo que estas ocorrem e, após o término dessa transmissão, o dispositivo final fica disponível para receber informações em duas janelas curtas, o que possibilita receber informações do *gateway*. Após essas comunicações, o dispositivo entra em modo de hibernação com o objetivo de economizar energia, sendo que o ciclo pode ser iniciado novamente caso ocorra novos eventos. (LORA ALLIANCE, 2019).

Classe B: Também operando no modo bidirecional, mas com a potência não tão reduzida quanto na classe anterior, herda ainda a funcionalidade assíncrona na presença de eventos, mas agora também opera em modo síncrono, uma vez que sincroniza com o *gateway* através de *beacons* periódicos, que tem o objetivo de anunciar a presença da rede sem fio, sincronizando com os dispositivos, abrindo janelas de recepção em tempos programados, possibilitando, assim, uma latência determinística e programável em até 128 segundos. Em contrapartida, com maior consumo de energia, uma vez que o rádio estará com a recepção disponível em uma maior parte do tempo. (LORA ALLIANCE, 2019).

Classe C: Como nas outras classes, opera em modo bidirecional, mas agora com menor latência, contemplando as mesmas especificações da Classe A, no quesito de transmissões aleatórias em eventos, mas agora, mantendo a recepção no dispositivo final sempre ligada, permitindo o *gateway* e os servidores se comunicarem com o dispositivo remoto sem a necessidade de sincronização de tempo ou então em janelas após a transmissão, o que faz com que o consumo de energia aumente muito, tornando inviável a utilização de baterias como fonte de alimentação. Em alguns casos, pode se transformar temporariamente um dispositivo classe A para C, com o intuito de atualização de firmware "*over the air*", sendo que, após a conclusão do processo, deve-se retornar para a classe A, caso a fonte principal de alimentação sejam baterias. (LORA ALLIANCE, 2019).

Com relação a segurança das informações que trafegam, o protocolo LoRaWAN possui duas camadas de segurança, uma na rede, que garante a autenticidade de todos os dispositivos finais e a outra que atua na parte de aplicação, garantindo que os dados que trafegam entre o *gateway* e o servidor de aplicação não sejam identificados. Já a criptografia utilizada é a do tipo AES (*Advanced Encryption Standard*), com a troca de chaves utilizando um identificador IEEE EUI-64 (*Extended Unique Identifier 64 bits*), sendo este o mesmo algoritmo utilizado em redes que utilizam o protocolo IPv6. (LORA ALLIANCE, 2019).

Conforme visto no tópico que finda, os protocolos DNP3, MQTT e LoRaWAN, serão utilizados no decorrer deste trabalho, sejam para que os objetivos sejam alcançados ou, então, apenas para prova de conceito, que de maneira indireta auxiliam na consolidação deste trabalho.

No tópico seguinte, serão apresentados alguns dispositivos de *hardware* que são necessários para que tudo que foi apresentado até agora seja colocado em prática.

2.6 Módulos utilizados

Uma vez que o desenvolvimento deste trabalho requer a aplicação de *hardwares*, optou-se pela pesquisa de módulos já prontos e disponíveis no mercado, de modo a alavancar o desenvolvimento. Assim, foi necessária uma pesquisa específica, levantando, de forma macro, as necessidades e chegando-se nos módulos seguintes como desejáveis no processo de formulação do protótipo.

2.6.1 Conversor DC-DC

Esses módulos possibilitam o rebaixamento de tensões DC (*Direct current*), provenientes de baterias ou, então, de fontes de alimentação, sendo elas chaveadas ou não. Um dos critérios para escolha dos módulos foi a eficiência de conversão, procurando assim, modelos com eficiência acima de 90%, o que indica que pouca potência será perdida na forma de calor dissipada pelos componentes dos módulos. Outro fator é a possibilidade de ajustar através de potenciômetro a tensão necessária na saída, para alimentação dos demais módulos.

No Quadro 5, pode-se observar um comparativo de módulos conversores DC-DC, que foram avaliados para o desenvolvimento deste trabalho.

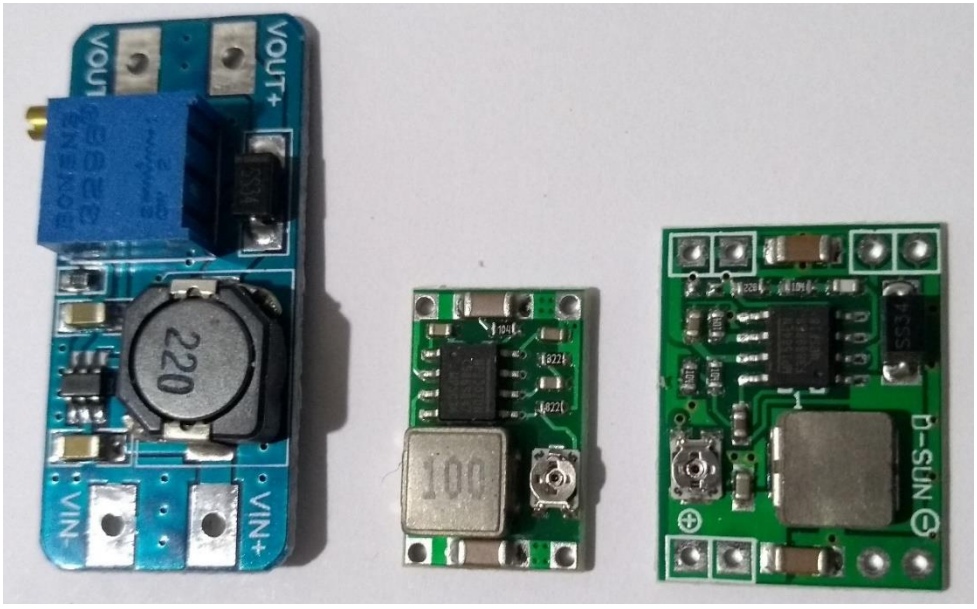
Quadro 5 – Comparativo módulos DC-DC

Conversores DC-DC	**LM2596**	***Ultra Small***	**Mini-360**
Tensão de entrada	3,2 V até 40 V	4,5 V até 28 V	4,75 V até 23 V
Tensão de saída	1,5 V até 35 V	0,8 V até 20 V	1 V até 17 V
Corrente saída	2 A	3 A	3 A
Eficiência de conversão	92 %	96 %	95 %
Temperatura operação	-40ºC até 85ºC	-45ºC até 85ºC	-40ºC até 85ºC
Frequência de operação	150 kHz	1,5 MHz	340 kHz
Dimensões	46 mm x 22 mm	22mm x 17mm	17 mm x 11 mm

Fonte: Elaborado pelo autor.

Na Figura 12, os três módulos DC-DC pesquisados para este trabalho são apresentados em ordem, sendo eles, LM2596, *Ultra Small* e Mini-360. Com eles, a parte de energização dos demais módulos poderá ocorrer sem problemas.

Figura 12 – Conversores DC-DC



Fonte: Elaborado pelo autor.

No tópico a seguir, será apresentado o conversor de nível de tensão para a interface serial, que é responsável pela interligação do módulo com equipamentos que operam em níveis de tensão diferentes. Convém ressaltar que a interface serial é importante, pois todas as informações passarão por esta interface.

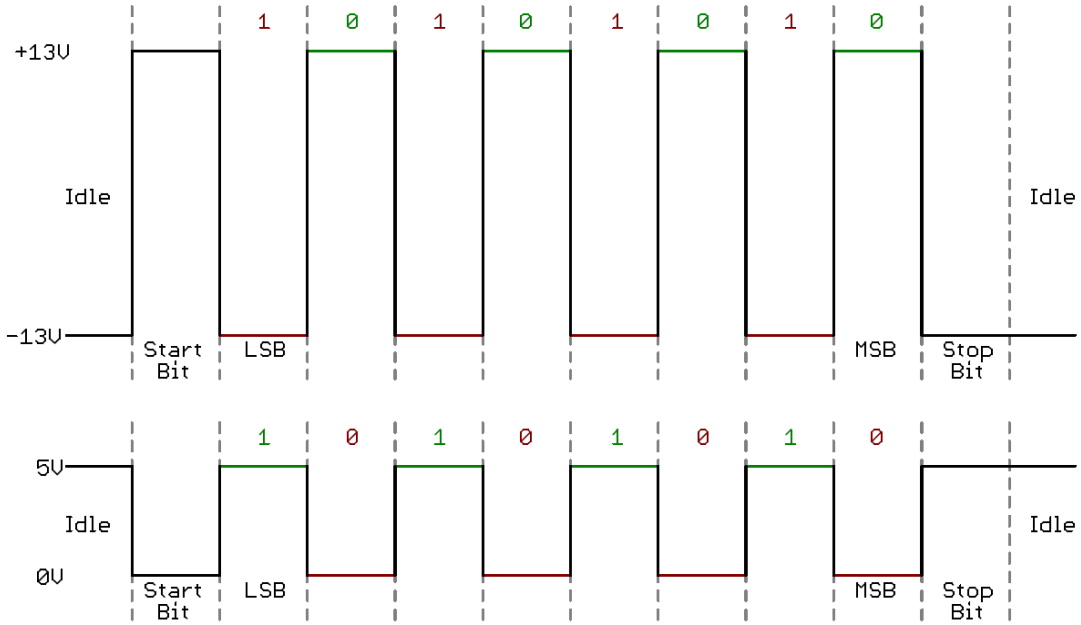
2.6.2 Conversor TTL para RS-232

Este módulo serve para converter o nível de tensão TTL (*Transistor-Transistor Logic*), que opera com níveis de tensão entre (0 V e 3,3 V) ou (0 V e 5 V), enquanto que, no padrão RS-232, opera com níveis de tensão entre (-25 V e +25 V) ou (-13 V e +13 V) e é usado na maioria dos computadores. (SPARKFUN, 2019).

Na Figura 13, pode-se verificar um comparativo entre os níveis de tensão de uma interface operando em TTL, que opera na faixa de 0 V até 5 V, tendo assim, uma amplitude de 5 V. Em contra ponto, observa-se também os níveis de tensão de uma interface operando em RS-232, que são muito maiores, ficando entre -13 V até 13 V, tendo assim, uma amplitude de 26 V. Portanto, para evitar danos aos circuitos e

dispositivos envolvidos no desenvolvimento deste trabalho, torna-se essencial a utilização deste conversor de nível de tensão.

Figura 13 – Comparativo níveis de tensão TTL e RS-232



Fonte: SPARKFUN (2019).

Na Figura 14, pode-se observar uma imagem do circuito responsável pela conversão de nível, normalmente utilizando-se apenas de um circuito integrado, fornecido por muitos fabricantes.

Figura 14 – Conversor TTL para RS-232

Fonte: Elaborado pelo autor.

O tópico seguinte apresentará uma ferramenta que auxiliará na realização de testes e medição de potência do módulo de rádio que será apresentado em breve.

2.6.3 Analisador de espectro - MSP-SA430-SUB1GHz

Analisador de espectro com interface de gerência de fácil utilização, contribuindo assim, para o desenvolvimento de sistemas de rádio frequência (RF) operando na faixa frequência ISM.

No Quadro 6, as características do módulo analisador de espectro são apresentadas. Convém ressaltar que a frequência de operação para os rádios neste trabalho será na faixa dos 915 MHz.

Quadro 6 – Principais características do MSP-SA430-SUB1GHz

Range de frequências de operação	300 a 348 MHz 389 a 464 MHz 779 a 928 MHz
Nível máximo de entrada	-40 dBm (típico)
Nível mínimo detectável	-100 dBm (típico)
Nível de resolução	0,5 dB
Alimentação	5 volts DC (USB 2.0)

Fonte: Texas Instruments (2019).

Na Figura 15, é apresentado o analisador de espectro. Trata-se de um dispositivo simples, alimentado através da porta USB (*Universal Serial Bus*) do computador, por onde as informações são coletadas através do *software* fornecido pelo próprio fabricante.

Figura 15 – Analisador de espectro MSP-SA430-SUB1GHz

Fonte: Elaborado pelo autor.

2.6.4 Módulo LoRa - HELTEC

Este módulo tem integrado, na mesma placa, um processador ESP32 e o módulo LoRa de rádio SX1276, podendo este operar nas frequências de 868 MHz e 915 MHz e conta com conector para bateria e antena, o que facilita bastante sua utilização, além de outras características que são listadas no Quadro 7.

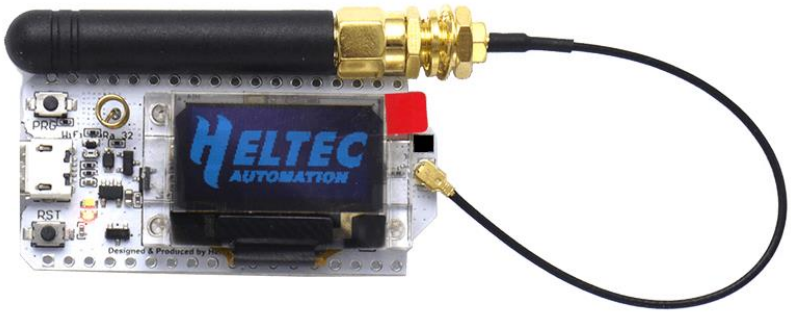
Quadro 7 – Principais características do módulo HELTEC

Processador	ESP32 (80 a 240 MHz *Tensilica* LX6 *dual-core*)
Interfaces	UART, SPI, I2C e Micro USB
Entradas/Saídas	Analógicas, Digitais e Display LCD
Memória RAM	448 Kb
Memória *Flash*	8MB
LoRa *Chip*	SX1276 (868 and 915 version)
Antena	Conector IPEX
Sensibilidade	-139 dBm (máxima)
Potência de TX	18dB ± 2dB
Modulação	LoRa, FSK, GFSK e OOK
Consumo	10,8 mA (RX) e 130 mA (TX)
Alimentação	5 Vdc ou 3,3 Vdc
Dimensões	50,2 mm x 25,5 mm
Temperatura	-40ºC a 85ºC

Fonte: HELTEC (2019).

Na Figura 16, pode-se observar a imagem do módulo LoRa.

Figura 16 – Módulo LoRa Heltec 915Mhz

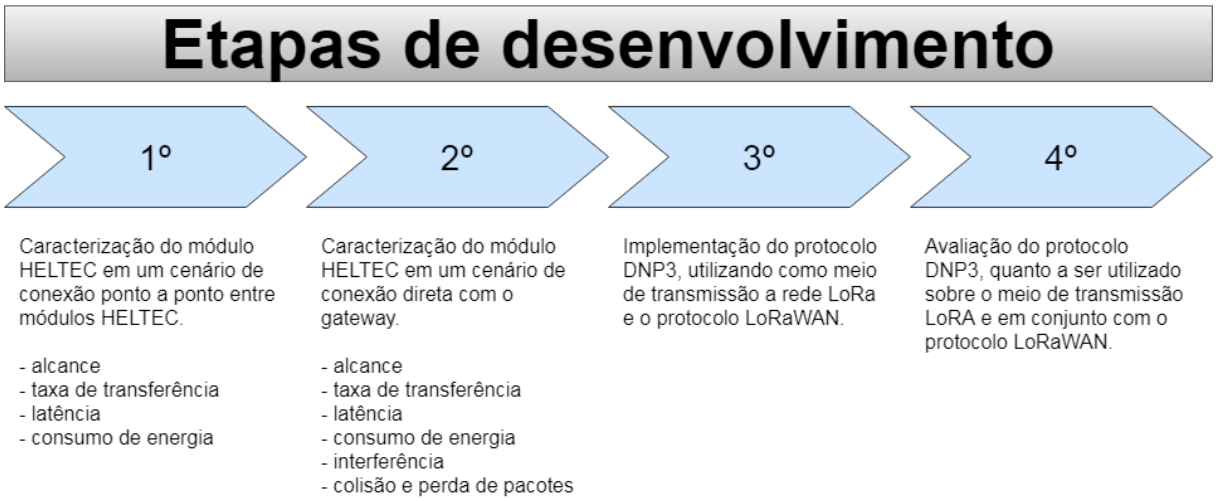


Fonte: HELTEC (2019).

3 METODOLOGIA

Neste capítulo, serão apresentadas as etapas desenvolvidas neste trabalho, iniciando pela montagem do protótipo, passando pela caracterização e desenvolvimento do *software*, além da realização de testes e validação final do conjunto *hardware* e *software*. A Figura 17 retrata, resumidamente, estas quatro etapas.

Figura 17 – Fluxograma das etapas de desenvolvimento



Fonte: Elaborado pelo autor.

Convém ressaltar que as etapas 1 e 2 são independentes das demais, mas as etapas 3 e 4 dependem diretamente da etapa 2.

3.1 Montagem do protótipo

Conforme visto no capítulo anterior, tratam-se de módulos desenvolvidos pelos seus respectivos fabricantes, ou seja, sem a necessidade de prototipação de circuitos impressos, tornando assim, esta etapa mais simples de ser elaborada.

Portanto, para que isso seja possível, foram verificados, após pesquisas, necessidade de *hardwares* específicos, mas devido ao escopo estar limitado a parte de comunicação, não iremos desenvolver *hardwares*, mas sim agrupar módulos necessários de modo que estes supram nossas necessidades.

Para a alimentação, temos a possibilidade de utilizar fontes com entrada AC, que transformam para 12 Vdc, o uso de baterias, ou, então, a utilização da própria

interface USB dos computadores. Isto é possível graças aos módulos LoRa serem desenvolvidos para baixo consumo de energia.

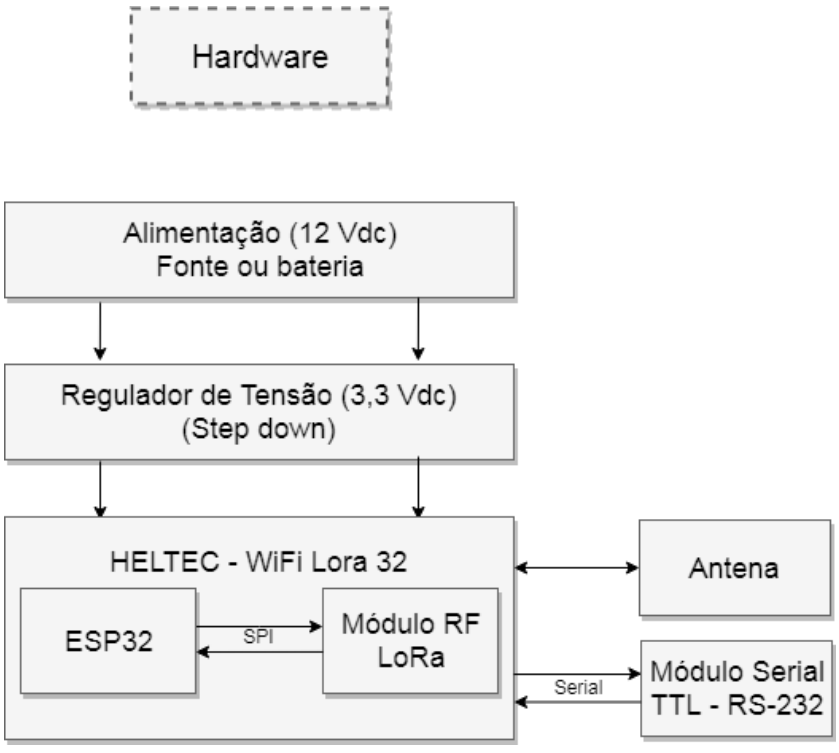
Uma vez utilizado o modo de alimentação com 12 Vdc, faz-se necessário a regulação dessa tensão para um valor menor para alimentação dos demais módulos. Como verificado durante a pesquisa, verificou-se que temos por unanimidade a utilização de 3,3 Vdc. Portanto, é possível a utilização de apenas um regulador para atender a demanda energética dos dois módulos.

Como optou-se pela utilização do módulo HELTEC, conforme visto anteriormente, este acaba por disponibilizar o processador ESP32 e chip LoRa SX1276, necessários para a realização deste trabalho, integrados na mesma placa.

Para que a comunicação se concretize, resta realizar a conversão da interface serial disponibilizada em nível TTL para RS232, sendo necessária a realização da conexão com o Módulo Conversor Serial TTL – RS-232, fazendo assim, a conversão de níveis de tensão entre a interface serial do microcontrolador e a interface serial do equipamento com protocolo DNP3.

Na Figura 18, podem ser vistos os blocos de *hardware*, que foram elencados como sendo essenciais para a montagem do protótipo.

Figura 18 – Visão dos blocos necessários para o protótipo



Fonte: Elaborado pelo autor.

Apresentados os detalhes referentes a montagem dos protótipos, no próximo tópico serão apresentados os detalhes relacionados aos testes necessários para o desenvolvimento deste trabalho.

3.2 Caracterização do módulo – Ponto a Ponto

Com o objetivo de caracterizar o módulo HELTEC ESP32, pretende-se realizar testes de transmissão de dados entre dispositivos de mesmo modelo, de modo a medir o alcance, taxa de transferência de dados, latência na transferência dos dados e o consumo de energia, uma vez que o objetivo dessas redes é a utilização em sistemas de baixo consumo de energia com alimentação por baterias, para proporcionar um longo período sem necessidade de intervenção.

3.2.1 Medição do alcance de transmissão

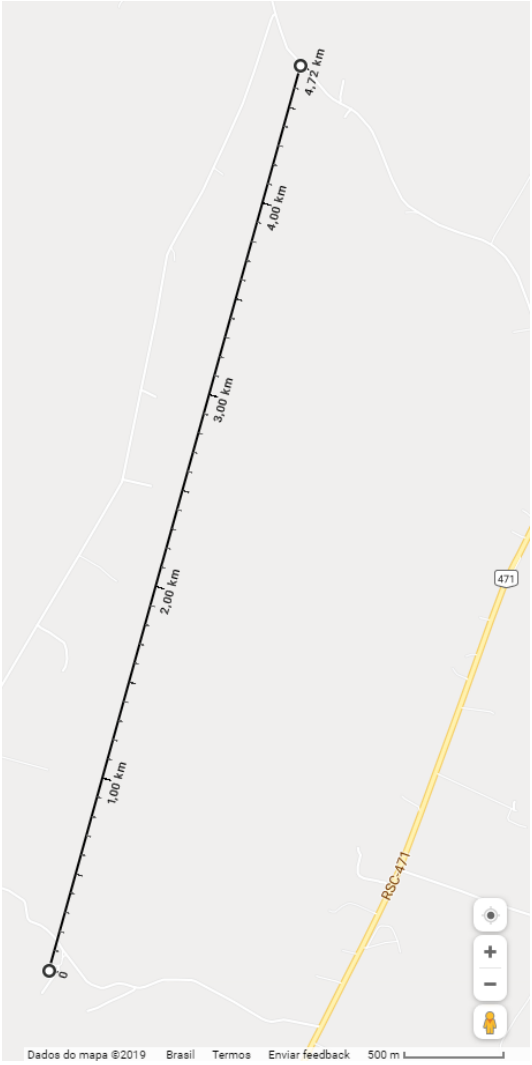
Como pretende-se caracterizar o módulo, é necessário que o local não tenha equipamentos transmitindo na faixa de frequência aberta em 915MHz. Portanto, optou-se pela realização dos testes em zona rural, onde, em muitos casos, é possível conseguir grandes distâncias com visada direta, o que impacta em melhores resultados, uma vez que não existem obstáculos que de alguma maneira possam interferir na comunicação dos dados. Com a utilização de aplicativos de mapas disponíveis para celular, pretende-se verificar a distância entre o transmissor que está em um ponto fixo até o receptor que pode estar em deslocamento. Inicialmente, estipulou-se uma distância maior que a especificada/recomendada pelo fabricante do módulo.

O local escolhido fica localizado no interior do município de Encruzilhada do Sul, pertencente ao estado do Rio Grande do Sul. Algumas áreas de terra na região são excelentes para este tipo de trabalho, uma vez que o relevo e a vegetação acabam por facilitar a visada entre os rádios LoRa responsáveis pela transmissão e recepção dos dados durante a execução dos testes.

Estipulou-se a distância de 4700 metros, maior que a especificada pelo fabricante, que é de aproximadamente 2800 metros, sendo recomendado para esta verificação a utilização da potência máxima de transmissão de 20 dBm (*decibel miliwatt*).

Na Figura 19, pode-se observar a região eleita. Portanto, onde os testes serão realizados, utiliza-se o local onde a contagem é igual a 0 metros, como o ponto onde ficará o transmissor.

Figura 19 – Imagem do *Google Maps* Encruzilhada do Sul – RS



Fonte: Encruzilhada do Sul (2019).

No tópico seguinte, serão apresentadas as métricas utilizadas para elaboração do teste de taxa de transferência.

3.2.2 Taxa de transferência

Para a realização deste teste, faz-se necessário a transmissão de algumas informações do módulo transmissor para o módulo receptor. Para isso, foi criado o padrão de cabeçalho apresentado no Quadro 8, que contem 14 *bytes*, com

informações que ajudam a identificar o pacote recebido, o tempo desde a inicialização do módulo que está realizando as transmissões e alguns delimitadores, de modo a tornar a informação fácil de ser interpretada durante a depuração. Maiores detalhes podem ser observados no Quadro 8.

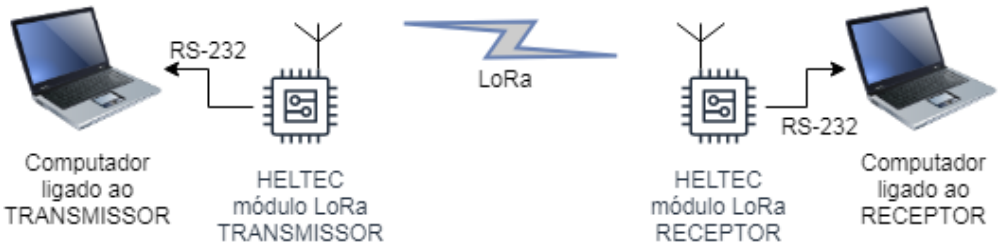
Quadro 8 – Formato da mensagem enviada

Delimitador	Identificador	Delimitador	Tempo	Delimitador	Final *string*
#	0 a 65535	#	00:00:00	#	\0
1 *byte*	2 *bytes*	1 *byte*	8 *bytes*	1 *byte*	1 *byte*

Fonte: Elaborado pelo autor.

Como visto no Quadro 8, a mensagem conterá sempre 14 *bytes*. Deste modo, resta apenas saber exatamente o momento em que a mensagem foi enviada. Para isso, foi introduzida uma mensagem de depuração, que é enviada pela serial do módulo transmissor, que por sua vez, está conectado a um computador, que armazenará estas informações. O mesmo processo deve ser realizado no módulo receptor, que deverá armazenar as informações recebidas no processo de depuração. Na Figura 20, verifica-se como essa infraestrutura foi projetada.

Figura 20 – Infraestrutura utilizada nos testes ponto a ponto



Fonte: Elaborado pelo autor.

Sabendo ainda que, inicialmente, os relógios de ambos os computadores ligados ao transmissor e receptor foram sincronizados na casa dos milésimos de segundo, podemos, de forma simples, calcular a taxa de transferência pela Equação 1.

$$Taxa\_de\_transferência = \frac{quantidade\_de\_bytes}{(momento_{TX} - momento_{RX})} \quad (1)$$

Sendo as unidades necessárias para o desenvolvimento desta equação apresentadas no Quadro 9:

Quadro 9 – Unidades para a taxa de transferência de dados

Taxa de transferência	B/s (*Bytes* por segundo)
Quantidade de *bytes*	*Byte*
Momento transmissão e recepção	segundo

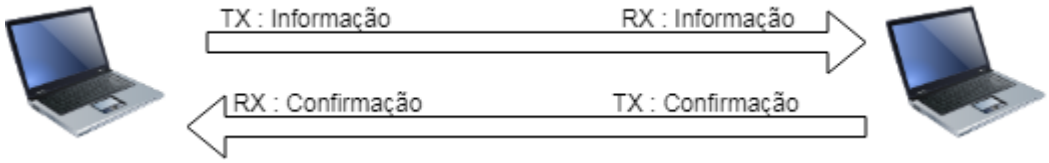
Fonte: Elaborado pelo autor.

Vistas, no presente tópico, as informações referentes ao teste de taxa de transferência, no tópico a seguir serão apresentadas as métricas para o teste de latência. Portanto, toda a infraestrutura montada e desenvolvida até o presente momento será aproveitada.

3.2.3 Latência

Para verificação da latência, deve ser considerado o momento em que a informação é transmitida pelo transmissor, recebida pelo receptor e respondida por este com a confirmação do recebimento, para o transmissor que iniciou todo o processo de comunicação, conforme pode ser visualizado na Figura 21.

Figura 21 – Exemplo de latência ideal



Fonte: Elaborado pelo autor.

Em resumo, esse seria o cenário ideal, onde o transmissor sabe o momento em que a mensagem foi enviada e o momento em que a confirmação é recebida. Essa diferença de tempos é a latência.

No caso deste trabalho, devido aos testes terem sido executados com o rádio LoRa no modo de operação “Classe A”, vamos considerar a latência como sendo o tempo da transmissão de dados do transmissor para o receptor multiplicada por dois. Este procedimento não é o ideal, mas dadas as circunstâncias, é a melhor situação. Então, deve-se considerar a seguinte Equação 2.

$$Latência = (momento_{TX} - momento_{RX}) * 2 \quad (2)$$

Realizados os testes relacionados a taxa de transferência e latência, faz-se necessária a validação referente ao consumo de energia, sendo isto apresentado em detalhes no tópico seguinte.

3.2.4 Consumo de energia

Conforme informado pelo fabricante em seu *datasheet*, o consumo operando em transmissão utilizando potência máxima de 20 dB (*Decibel*) é de 130 mA (*mili ampere*). Para confirmar isso, será necessária a intervenção com multímetro na escala de mA para medição da corrente de alimentação do módulo LoRa.

Já no módulo em operação no modo de recepção, segundo informado pelo fabricante, o consumo deve ser na faixa dos 10 mA.

Com os testes realizados até o presente momento, a caracterização dos módulos em configuração ponto a ponto está encerrada. Portanto, no tópico a seguir, será explorada a configuração módulo – *gateway*, sendo necessárias algumas alterações significativas em termos de *software* e *hardware*, com a utilização de um *gateway* que concentrará os pacotes transmitidos pelo módulo.

3.3 Caracterização do módulo – *Gateway*

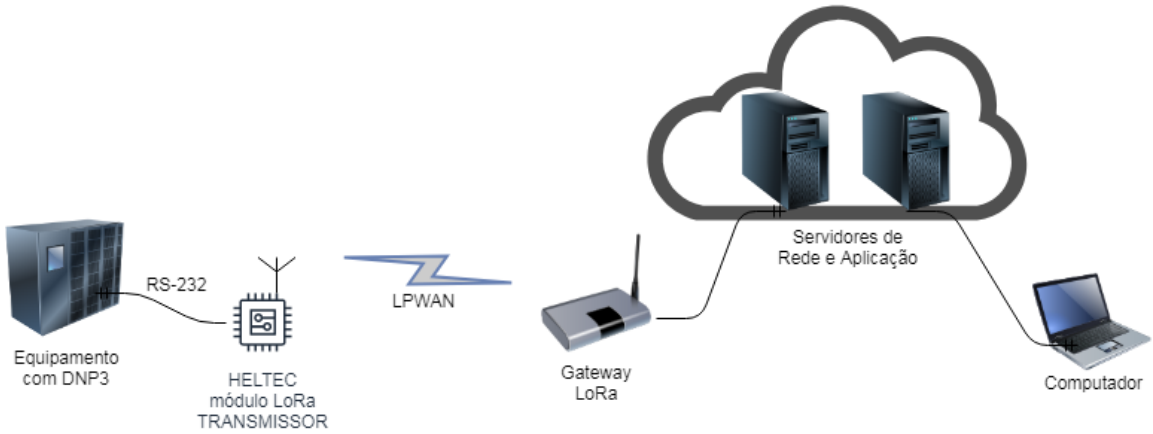
Basicamente, trata-se de alguns itens expostos no capítulo anterior, sendo eles as taxas de transferência e latência. Além destes, será analisada a interferência de outros dispositivos que estejam utilizando a mesma frequência e testes, confirmando a integridade das informações transmitidas pelo módulo HELTEC LoRa que são recebidas no servidor de aplicação.

Com relação ao *gateway*, servidor de Rede e Aplicação, estas estruturas já foram previamente implementadas, cabendo ao presente trabalho apenas utilizá-las, adaptando-se a alguns parâmetros de configuração que serão explicados nos momentos oportunos.

3.3.1 Taxa de transferência

Neste caso, a taxa de transferência se diferencia do caso anterior, no aspecto da estrutura da rede, onde os módulos em teste estarão conectados a um *gateway*, sendo este o concentrador de outros módulos LoRa, englobando assim, uma rede composta por mais dispositivos e, consequentemente, exigindo a troca do cenário envolvido, conforme pode ser visto na Figura 22.

Figura 22 – Taxa de transferência com o uso do *Gateway* LoRa



Fonte: Elaborado pelo autor.

A seguir, será apresentado, como serão realizados os testes de latência na configuração módulo – *gateway*, diferenciando-se em alguns aspectos da configuração ponto a ponto.

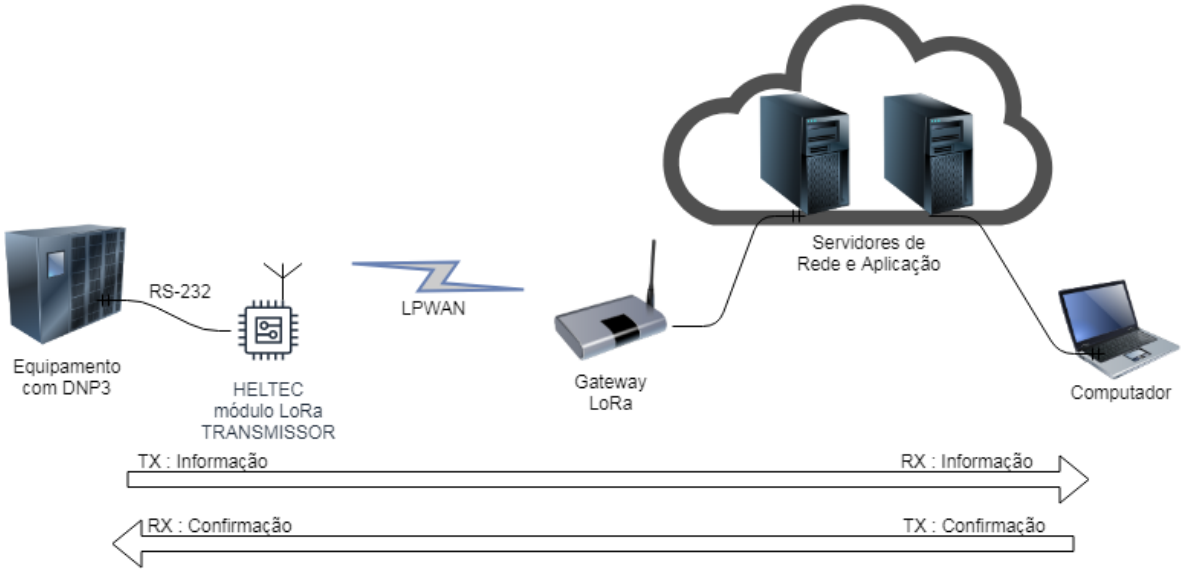
3.3.2 Latência

A grande mudança se dará neste teste, pois além da mudança no aspecto da estrutura da rede, onde os módulos em teste estarão conectados a um *gateway*, sendo este o concentrador de outros módulos LoRa, agora haverá a utilização da operação do rádio em Classe C, conforme visto no tópico de Protocolos, possibilitando assim, a resposta de uma informação enviada, conforme pode ser visto na Figura 23.

Cabe lembrar que estará sendo utilizado o recurso de confirmação de recebimento de pacotes por parte do transmissor e pelo *gateway* LoRa. Desse modo, todo pacote enviado que esteja corretamente formado, terá uma confirmação de recebimento enviada de volta ao transmissor inicial da comunicação, podendo assim, obter-se uma maior precisão com relação a latência da rede.

Na Figura 23, se apresenta a configuração mais completa para a realização de testes, visto que, agora, tem-se por objetivo começar o processo para externar as informações para que estas possam ser acessadas através da *Internet*, fechando o ciclo fundamental para a concretização de mais um objetivo do presente trabalho.

Figura 23 – Latência com o uso do *Gateway* LoRa



Fonte: Elaborado pelo autor.

No tópico seguinte, será explorada a questão de interferência, visto que, como trata-se de equipamentos que exploram frequências abertas, podem existir outros equipamentos que poderiam, de alguma maneira, influenciar na comunicação entre o módulo e o *gateway*.

3.3.2 Interferência rádios LoRa

Outro teste a ser realizado é o de interferência na parte de rádio. Neste caso, com o auxílio de um analisador de espectro, empregado para análise de interferências, se verifica a presença de outros equipamentos utilizando a mesma frequência e canais do utilizado no módulo desenvolvido neste trabalho.

Visto a questão da interferência de outros equipamentos, no tópico a seguir, será explorada a integridade dos dados, visto ser necessário que a informação transmitida pelo módulo e a recebida no servidor de aplicação não sejam alteradas, de maneira alguma, pelo meio de transmissão.

3.3.3 Integridade dos pacotes

Este teste tem por objetivo verificar a integridade dos *bytes* recebidos, de modo a validar que o dado transmitido pelo servidor de aplicação é o mesmo dado que está chegando no equipamento com o protocolo DNP3. Para o sentido inverso, essa análise também será válida.

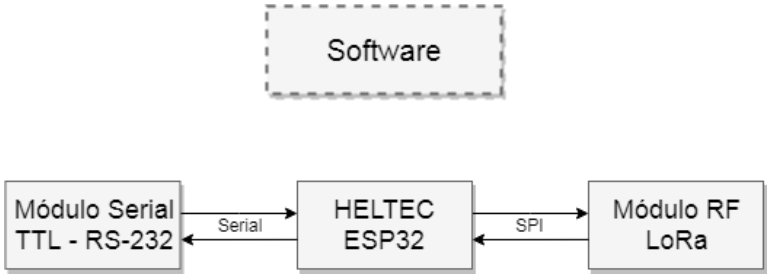
Realizados os testes de taxa de transferência, latência, interferência e integridade dos dados, encerra-se mais um dos principais objetivos do presente trabalho. Assim, no tópico seguinte, serão apresentados os passos necessários para o transporte da informação gerada pelo protocolo DNP3 até o seu destino no servidor de aplicação.

3.4 Implementação de *software* para transporte do DNP3 sobre o LoRaWAN

Este item claramente é um dos mais importantes do trabalho, pois tem por objetivo desenvolver todo o código fonte que será embarcado no módulo HELTEC ESP32, que foi apresentado no tópico anterior, conectando assim, através de interface serial, o equipamento que tem suporte ao protocolo DNP3.

De forma macro, a parte do *software* a ser desenvolvida, será com relação ao registro do módulo LoRa no *gateway*, assim como todo o protocolo LoRaWAN que roda para fechar o enlace de rádio. Mais baixo nível, temos a comunicação SPI (*Serial Peripheral Interface*) entre o microcontrolador e o módulo LoRa para fechar a comunicação com o equipamento através da serial, uma vez que se tem por objetivo apenas transportar o protocolo DNP3 de modo transparente sobre o protocolo LoRaWAN, conforme pode-se observar na Figura 24.

Figura 24 – Visão dos blocos necessários no *software*.



Fonte: Elaborado pelo autor.

Deve-se trabalhar com interrupções em ambos os lados, de modo a identificar a chegada de dados, pois como visto anteriormente, o protocolo DNP3 opera com o dispositivo escravo (*outstation*) enviando informações sem a solicitação do mestre (*master*) da rede. *Buffers* também são necessários, uma vez que grandes quantidades de dados podem trafegar por um longo período, visto que nos módulos LoRa, temos uma baixa taxa de transferência, com objetivo de alcançar a grandes distâncias com baixo consumo de energia.

Após a implementação do *software* para transporte do DNP3 sobre LoRaWAN, encerra-se mais um objetivo, fazendo-se agora necessária a avaliação referente as informações que foram transmitidas por este meio. Portanto, no tópico a seguir, serão apresentadas tais métricas de avaliação das informações.

3.5 Avaliação do protocolo DNP3 sobre o LoRaWAN

Realização de testes comparativos entre uma conexão através de interface RS-232 (direta) ao equipamento com DNP3 e a uma conexão utilizando toda a infraestrutura da rede LoRa.

Primeiro teste: Utilizando um computador com interface serial RS-232, conectar-se ao equipamento que dispõe do protocolo DNP3, realizando a consulta de alguns parâmetros por comandos enviados a este equipamento, medindo o tempo de resposta e se estas estão corretas, ou seja, não tendo sido corrompidas na transmissão. Na Figura 25, pode ser vista de forma ilustrada a forma de conexão a ser realizada.

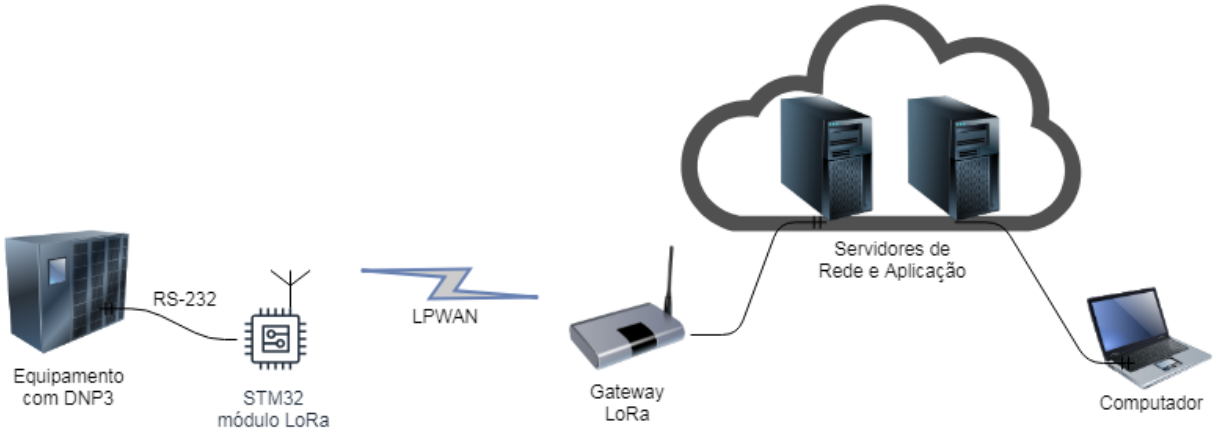
Figura 25 – Teste comunicação protocolo DNP3 utilizando computador.

Fonte: Elaborado pelo autor.

Segundo teste: Com os resultados do teste acima, realizar novamente a consulta de parâmetros enviando comandos ao equipamento, mas agora utilizando-se, como meio de transmissão, o módulo LoRa, que se conecta ao concentrador

(*gateway*), o servidor de rede e da nuvem disponível na topologia LPWAN. Na Figura 26, pode ser vista de forma ilustrada a forma de conexão a ser realizada.

Figura 26 – Teste comunicação utilizando *gateway* LoRa



Fonte: Elaborado pelo autor.

Com isto, finda-se a metodologia, onde foram apresentadas as etapas desenvolvidas. Desta forma, no capítulo seguinte, serão apresentados os resultados obtidos durante a realização deste desenvolvimento.

4 ANÁLISE DOS RESULTADOS

Neste capítulo serão apresentados os resultados dos testes e implementações realizadas durante o desenvolvimento deste trabalho e previamente descritas no tópico de Metodologia e nos objetivos específicos.

4.1 Montagem do protótipo

Conforme anteriormente explicado, na montagem dos protótipos não houve problemas. Inicialmente planejava-se a utilização de outros modelos de módulos, mas achou-se ideal focarmos apenas no do fabricante HELTEC. Este fato pode possibilitar a exploração de outros modelos em outros cenários que exijam um alcance maior.

Dado o sucesso da montagem do protótipo, serão apresentados agora os resultados dos testes para caracterização do módulo, conforme pode ser visto no tópico seguinte.

4.2 Caracterização do módulo – Ponto a Ponto

A caracterização transcorreu sem maiores problemas, pois inicialmente, foi realizada em ambiente de testes, o que possibilitou validar com bastante eficiência os procedimentos que seriam realizados em campo, uma vez que, estando no local dos testes, algumas alterações poderiam ser difíceis de realizar, dado o tempo que se definiu para estas atividades.

4.2.1 Medição do alcance de transmissão

Com relação ao alcance do *link* de rádio, conseguimos atingir 3400 metros (3,4 Quilômetros), distância maior que a especificada pelo fabricante utilizando as antenas do tipo toco, que são fornecidas juntamente com o módulo. Existem relatos, no fórum do fabricante, de pessoas que conseguiram transmissões maiores que 10 km (Quilômetros), com antenas que fornecem um ganho maior, mas para o que queríamos provar com esta atividade, surpreendeu positivamente o resultado exposto.

O módulo transmissor foi instalado em uma altura de aproximadamente 8 metros em relação ao solo. Com o auxílio de um computador, foi possível realizar a análise dos pacotes transmitidos, possibilitando assim o cruzamento das informações.

Já com relação ao módulo receptor, este foi fixado na parte externa de um veículo, com o qual inicialmente obteve-se uma melhor eficiência nos testes e, numa primeira análise, para se verificar o maior alcance mais rapidamente.

Na Figura 27, é apresentada a imagem do mapa, onde foi realizado o teste de alcance. Convém lembrar que o marco “0” (zero) é onde o transmissor foi instalado. Enquanto isso, o receptor se deslocava até a marca dos 3,4 km.

Figura 27 – Imagem do *Google Maps* Encruzilhada do Sul – RS - Alcance real

Fonte: Encruzilhada do Sul (2019).

Do local onde se conseguiu essa recepção, era possível enxergar o local onde o módulo transmissor estava fixado, fator este que possibilitou tal distância alcançada.

Com a distância máxima estabelecida, no próximo tópico serão realizados os testes para definição da taxa de transferência na configuração ponto a ponto.

4.2.2 Taxa de transferência

Conforme explicado na metodologia, utilizou-se dois computadores para realizar a análise dos momentos em que os pacotes eram transmitidos e recebidos. Antes disso, os relógios desses computadores foram sincronizados com ajuda de um servidor NTP (*Network Time Protocol*), de modo a terem exatamente a mesma hora, assim como a contagem em milésimos de segundo. Dadas essas informações, obteve-se as seguintes informações para depuração:

No Quadro 10, são apresentadas as informações de depuração, tanto do módulo responsável pela transmissão quanto do módulo receptor. Convém ressaltar que os relógios de ambos computadores estavam sincronizados.

Quadro 10 – Informações de depuração do módulo transmissor e receptor

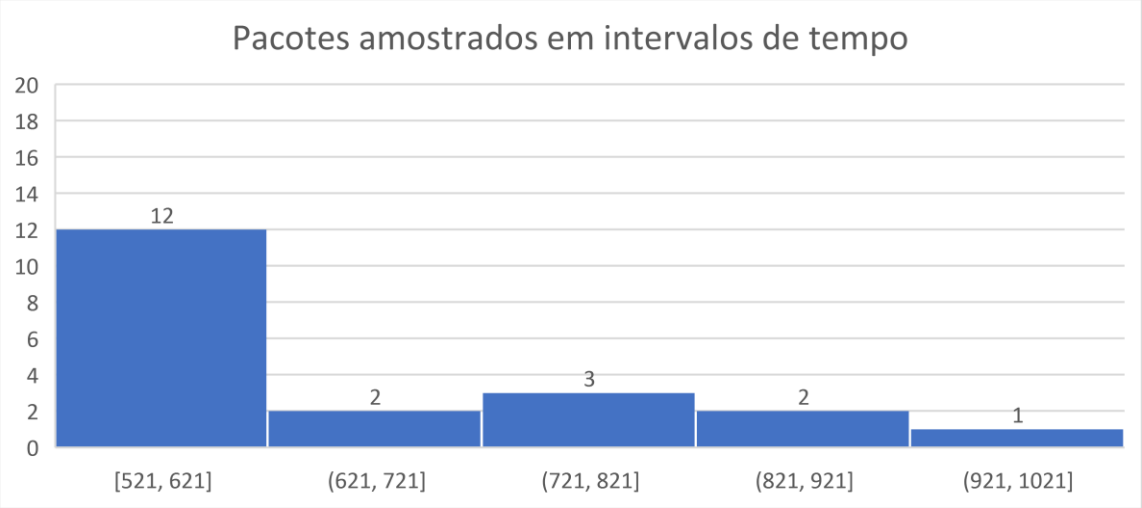
Módulo de TRANSMISSÃO:	Módulo de RECEPÇÃO
13:30:36.575 -> [#501#00:21:38#]	13:30:37.406 -> [#501#00:21:38#] RSSI=-114 SNR=11.75
13:30:39.172 -> [#502#00:21:41#]	13:30:39.739 -> [#502#00:21:41#] RSSI=-114 SNR=12.75
13:30:41.752 -> [#503#00:21:43#]	13:30:42.440 -> [#503#00:21:43#] RSSI=-114 SNR=13.00
13:30:44.362 -> [#504#00:21:46#]	13:30:44.992 -> [#504#00:21:46#] RSSI=-115 SNR=12.25
13:30:46.927 -> [#505#00:21:49#]	13:30:47.538 -> [#505#00:21:49#] RSSI=-114 SNR=13.00
13:30:49.507 -> [#506#00:21:51#]	13:30:50.054 -> [#506#00:21:51#] RSSI=-115 SNR=13.00
13:30:52.088 -> [#507#00:21:54#]	13:30:52.687 -> [#507#00:21:54#] RSSI=-115 SNR=12.75
13:30:54.652 -> [#508#00:21:56#]	13:30:55.615 -> [#508#00:21:56#] RSSI=-114 SNR=12.75
13:30:57.278 -> [#509#00:21:59#]	13:30:57.844 -> [#509#00:21:59#] RSSI=-115 SNR=12.75
13:30:59.860 -> [#510#00:22:02#]	13:31:00.657 -> [#510#00:22:02#] RSSI=-115 SNR=12.75
13:31:02.407 -> [#511#00:22:04#]	13:31:02.990 -> [#511#00:22:04#] RSSI=-114 SNR=12.25
13:31:05.006 -> [#512#00:22:07#]	13:31:05.730 -> [#512#00:22:07#] RSSI=-115 SNR=12.25
13:31:07.607 -> [#513#00:22:09#]	13:31:08.128 -> [#513#00:22:09#] RSSI=-114 SNR=13.25
13:31:10.173 -> [#514#00:22:12#]	13:31:10.782 -> [#514#00:22:12#] RSSI=-115 SNR=13.00
13:31:12.761 -> [#515#00:22:14#]	13:31:13.310 -> [#515#00:22:14#] RSSI=-114 SNR=12.75
13:31:15.344 -> [#516#00:22:17#]	13:31:15.912 -> [#516#00:22:17#] RSSI=-115 SNR=12.75
13:31:17.941 -> [#517#00:22:20#]	13:31:18.847 -> [#517#00:22:20#] RSSI=-115 SNR=12.50
13:31:20.521 -> [#518#00:22:22#]	13:31:21.082 -> [#518#00:22:22#] RSSI=-115 SNR=12.75
13:31:23.100 -> [#519#00:22:25#]	13:31:23.915 -> [#519#00:22:25#] RSSI=-115 SNR=13.00
13:31:25.665 -> [#520#00:22:27#]	13:31:26.233 -> [#520#00:22:27#] RSSI=-115 SNR=12.75

Fonte: Elaborado pelo autor.

Ao analisar as 20 amostras de pacotes recebidos, obteve-se um tempo médio da transmissão até a recepção dos 14 *bytes* de aproximadamente 660 ms (Milésimo de segundo) e, consequentemente, uma taxa de transferência média de 21,21 B/s, ficando um pouco abaixo da taxa mínima de 37,5 B/s, conforme visto no capítulo de fundamentação teórica, presente neste trabalho.

No histograma da Figura 28, pode-se observar a quantidade de pacotes em intervalos de tempo de 100 ms, apresentando-se em mais de 50% dos casos pacotes entre 521 e 621 ms.

Figura 28 – Pacotes recebidos agrupados por intervalos de tempo



Fonte: Elaborado pelo autor.

Analisada a questão da taxa de transferência, no tópico seguinte serão apresentados os resultados referentes a latência na configuração ponto a ponto.

4.2.3 Latência

Conforme visto no item anterior, obteve-se um tempo médio de transferência entre o módulo transmissor e o receptor de 660 ms. Considerando que isto é apenas a transferência de pacotes sem a confirmação do recebimento, considerou-se que se levara o mesmo tempo para a confirmação. Portanto, considera-se uma latência de aproximadamente 1320 ms (aproximadamente 1,3 segundos).

Apresentados os dados referentes ao teste de latência, no próximo tópico serão analisados os resultados referentes ao consumo de energia.

4.2.4 Consumo de energia

Um item importante ao realizar a caracterização do módulo é a corrente consumida por este. De acordo com o fabricante, o consumo médio é 130 mA. Durante a realização dos testes, verificou-se que o consumo chegou próximo dos 200 mA, conforme pode ser visto na Figura 29. Convém lembrar que estávamos com a serial habilitada devido ao modo de depuração e, também, com o *display* OLED (*Organic Light-Emitting Diode*) em uso.

Figura 29 – Corrente no módulo transmissor

Fonte: Elaborado pelo autor.

Já a corrente no momento em que o módulo receptor recebe as informações é de aproximadamente 66,1 mA, conforme pode-se observar na Figura 30. Uma observação em relação a corrente de recepção é que esta informação não é informada pelo fabricante em seu manual.

Figura 30 – Corrente no módulo receptor

Fonte: Elaborado pelo autor.

Estas leituras se referem ao momento em que houve o maior consumo de corrente, portanto, convém lembrar que isto não ocorre em 100% do tempo, variando conforme o modo de operação e frequência das transmissões.

Concluída a apresentação dos resultados referentes aos testes de alcance, taxa de transferência, latência e consumo de energia para a configuração ponto a ponto, no tópico seguinte serão apresentados os resultados para a configuração módulo – *gateway*.

4.3 Caracterização do módulo – *Gateway*

Durante a caracterização do módulo em conjunto com o *gateway*, não houve problemas, apenas sendo possível verificar que o comportamento é bem diferente de quando utilizado na topologia ponto a ponto, sendo que, em alguns momentos, a duração da transmissão dos pacotes ocorria em um tempo maior, fato este que será melhor explicado nos itens a seguir.

4.3.1 Taxa de transferência

Conforme já informado na metodologia deste trabalho, este teste baseia-se no tempo de envio de um pacote de tamanho conhecido entre o módulo transmissor, passando pelo *gateway* e chegando no servidor de aplicação.

Para que tal feito fosse possível de ser realizado, optou-se pela utilização de um mesmo computador para registrar as informações de depuração com a mesma base de tempo.

Alguns ajustes tiveram de ser realizados, pois a biblioteca fornecida pelo fabricante ainda está em desenvolvimento. As alterações se concentraram na questão do *Adaptive Data Rate* (ADR), sendo este desabilitado para que se pudesse realizar o envio de pacotes maiores e alterar a forma de varredura para recepção de pacotes, pois inicialmente, a biblioteca acaba retransmitindo muitas vezes a mesma mensagem, gerando assim muitas confirmações (ACK - *Acknowledgemet*) de modo desencontrado. Estas alterações foram necessárias, visto que o algoritmo que realiza o cálculo para definição de “velocidade” para transmitir os dados é extremamente complexo, pois considera inúmeras informações sobre a intensidade do sinal RSSI (*Received Signal Strength Indication*), quantidade de canais em uso, potência de transmissão, frequência de operação, entre outas informações.

No Quadro 11, são apresentadas as informações de depuração, de forma resumida, referentes a 20 amostras de transmissões de dados utilizando 14 *bytes* de informação, igualmente ao dos testes no modo ponto a ponto.

Quadro 11 – Informações de depuração do módulo transmissor

Módulo Transmissor	Servidor de Aplicação – MQTT
20:07:06.022 -> PACKET (1) SEND	20:07:06,306 : messageArrived() added: message #1
20:07:07.111 -> ACK RECEIVED	
20:07:36.649 -> PACKET (2) SEND	20:07:36,931 : messageArrived() added: message #2
20:07:37.736 -> ACK RECEIVED	
20:08:07.670 -> PACKET (3) SEND	20:08:07,888 : messageArrived() added: message #3
20:08:08.666 -> ACK RECEIVED	
20:08:38.585 -> PACKET (4) SEND	20:08:38,854 : messageArrived() added: message #4
20:08:39.673 -> ACK RECEIVED	
20:09:08.410 -> PACKET (5) SEND	20:09:08,697 : messageArrived() added: message #5
20:09:09.498 -> ACK RECEIVED	
20:09:38.686 -> PACKET (6) SEND	20:09:38,972 : messageArrived() added: message #6
20:09:39.773 -> ACK RECEIVED	
20:10:09.436 -> PACKET (7) SEND	20:10:09,723 : messageArrived() added: message #7
20:10:10.524 -> ACK RECEIVED	
20:10:39.627 -> PACKET (8) SEND	20:10:39,914 : messageArrived() added: message #8
20:10:40.731 -> ACK RECEIVED	
20:11:10.632 -> PACKET (9) SEND	20:11:10,923 : messageArrived() added: message #9
20:11:11.720 -> ACK RECEIVED	
20:11:41.370 -> PACKET (10) SEND	20:11:41,664 : messageArrived() added: message #10
20:11:42.463 -> ACK RECEIVED	
20:12:11.147 -> PACKET (11) SEND	20:12:11,416 : messageArrived() added: message #11
20:12:12.216 -> ACK RECEIVED	
20:12:40.735 -> PACKET (12) SEND	20:12:41,034 : messageArrived() added: message #12
20:12:41.811 -> ACK RECEIVED	
20:13:11.157 -> PACKET (13) SEND	20:13:11,451 : messageArrived() added: message #13
20:13:12.255 -> ACK RECEIVED	
20:13:40.610 -> PACKET (14) SEND	20:13:40,903 : messageArrived() added: message #14
20:13:41.697 -> ACK RECEIVED	
20:14:11.061 -> PACKET (15) SEND	20:14:11,344 : messageArrived() added: message #15
20:14:12.123 -> ACK RECEIVED	
20:14:41.274 -> PACKET (16) SEND	20:14:41,561 : messageArrived() added: message #16
20:14:42.328 -> ACK RECEIVED	
20:15:12.437 -> PACKET (17) SEND	20:15:12,719 : messageArrived() added: message #17
20:15:13.536 -> ACK RECEIVED	
20:15:43.790 -> PACKET (18) SEND	20:15:44,085 : messageArrived() added: message #18
20:15:44.880 -> ACK RECEIVED	
20:16:13.691 -> PACKET (19) SEND	20:16:13,980 : messageArrived() added: message #19
20:16:14.779 -> ACK RECEIVED	
20:16:44.720 -> PACKET (20) SEND	20:16:45,018 : messageArrived() added: message #20
20:16:45.808 -> ACK RECEIVED	

Fonte: Elaborado pelo autor.

Para determinação e cálculo da taxa de transferência, é necessário medir o tempo que o pacote levou do transmissor até o servidor de aplicação, sendo este de aproximadamente 283,7 ms. Considerando, neste caso, que o pacote continha 14 *bytes*, para que seja feito um comparativo com a topologia ponto a ponto, obteve-se uma taxa de transferência de 49,46 B/s, ficando acima da taxa mínima 37,5 B/s, conforme visto no capítulo de fundamentação teórica, presente neste trabalho.

Analisada a questão da taxa de transferência, no tópico seguinte serão apresentados os resultados referentes a latência na configuração módulo - *gateway*.

4.3.2 Latência

Conforme visto na metodologia deste trabalho, na comunicação com o *gateway*, uma mensagem de confirmação da recepção dos dados é enviada de volta ao transmissor, tornando viável a medição, sem a necessidade de aproximações ou considerações, conforme o item da transmissão ponto a ponto.

Elucidado o fato acima e conforme visto no item anterior, onde são expostas as informações de depuração, chegou-se ao valor médio de latência de 1062,95 ms (um pouco mais de 1 segundo). Sendo assim, encontrou-se um valor médio de latência de 1380 ms (1,3 segundos).

Apresentados os dados referentes ao teste de latência, no próximo tópico serão analisados os resultados referentes a interferência de outros equipamentos que possam estar operando na mesma frequência.

4.3.3 Interferência rádios LoRa

Utilizando a ferramenta de análise de espectro, durante a realização dos testes de Taxa de Transferência e Latência, não foram observados outros equipamentos utilizando a faixa de frequência de 916,8 MHz e 923,3 MHz, onde estas frequências se referem, respectivamente, a transmissão e a recepção do módulo LoRa HELTEC.

Visto que não houve interferências, no próximo tópico, serão apresentadas as considerações referentes a integridade das informações enviadas.

4.3.4 Integridade dos pacotes

Com relação a integridade dos dados, sempre que o pacote chegou no servidor de aplicação, o mesmo estava com sua informação igual ao momento em que foi gerada, ou seja, não houve troca de bits durante o caminho e mudanças de meio de comunicação.

Convém ressaltar que as informações que trafegaram estavam codificadas. Deste modo, foram validadas em seu formato codificado, sendo apenas algumas amostras decodificadas para efeito de validação, pois nos pacotes subsequentes as informações se repetiam.

Concluídos os testes para caracterização na configuração módulo – *gateway*, com a realização dos testes de taxa de transferência, latência, interferência e integridade dos dados, no tópico seguinte serão apresentadas as questões relacionadas a implementação para o transporte do protocolo DNP3 sobre o protocolo LoRaWAN.

4.4 Implementação de *software* para transporte do DNP3 sobre o LoRaWAN

Encerradas as etapas de caracterização do módulo, partiu-se para o desenvolvimento do *software* responsável pelo envio e recebimento de informações, uma vez que se optou pela operação em Classe C, onde o rádio envia informações quando necessário e as recebe, sem a necessidade de estar em janelas específicas de tempo.

De modo a iniciar o desenvolvimento de *software*, uma vez escolhido o módulo HELTEC ESP32 LoRa, optou-se pela utilização de uma biblioteca fornecida pelo próprio fabricante e mantida pela comunidade de desenvolvedores na *Internet*. Esta biblioteca pode ser obtida em (GITHUB HELTEC, 2019). A biblioteca traz alguns exemplos mais comuns de utilização, tais como na operação em Classe A. Não sendo este o nosso objetivo de operação, percebe-se a necessidade de desenvolver algumas partes especificas para Classe C.

4.4.1 Registro e conexão com o *Gateway*

Após a execução de alguns testes exploratórios utilizando os exemplos, verificou-se a necessidade de realizar ajustes para que o módulo conseguisse registrar-se e, consequentemente, conectar-se ao *gateway*, de modo que alterações tiveram de ser realizadas na biblioteca, sendo estas expostas no Quadro 12.

Quadro 12 – Principais alterações na biblioteca para registro e conexão

Frequência de operação	915Mhz
Definição de classe	*Class* C
Ativação Personalizada	Utilizando modo ABP (*Activation by Personalization*)
Endereço do dispositivo	0xAB008686
Chave de rede LoRaWAN	{0xAB, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}
Chave de Aplicação LoRaWAN	{0xAB, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}
Taxa de dados adaptável	Desabilitada
Tempo entre envios (ms)	5000

Fonte: Elaborado pelo autor.

Convém ressaltar a necessidade de desabilitar o uso da taxa de dados adaptável, uma vez que esta mostrou-se um limitador para o sucesso no envio de pacotes de tamanhos maiores.

Realizadas as alterações mencionadas anteriormente, o módulo se registrou corretamente no *gateway*, conseguindo transmitir dados previamente estipulados no código fonte para o servidor de aplicação, conforme pode ser visto no Quadro 13.

Quadro 13 – Dados transmitidos do módulo e recebidos no servidor de aplicação

Dados definidos no código fonte do módulo transmissor	Dados recebidos no servidor de aplicação codificada em Base64
1234567890ª	MTIzNDU2Nzg5MGE=

Fonte: Elaborado pelo autor.

Atingido o marco acima, partiu-se para a realização de testes no sentido oposto, onde o módulo HELTEC recebe as informações enviadas pelo servidor de aplicação.

No modo “recepção de dados”, o módulo não deve entrar em modo de economia de energia, estando sempre apto a receber informações. Sabendo-se de tal fato e utilizando-se do servidor de aplicação, uma mensagem contendo informações foi enviada ao módulo HELTEC, conforme apresentado no Quadro 14.

Quadro 14 – Dados transmitidos do servidor de aplicação e recebidos no módulo

Dados enviados do servidor de aplicação codificada em Base64	Dados recebidos no módulo LoRa
dW5pc2lub3M=	Unisinos

Fonte: Elaborado pelo autor.

Dado o exposto acima, concluiu-se que, de forma trivial, o envio e o recebimento de informações está ocorrendo, ainda que não seja com os dados que realmente deseja-se trabalhar. Porém, isto garante que a comunicação entre módulo HELTEC, *gateway* e servidor de aplicação está operacional, podendo assim, continuar com os ajustes para que se possa realizar a troca das informações desejadas.

4.4.2 Transmissão de pacotes com informações do protocolo DNP3

Como o objetivo é a comunicação de um equipamento de energia que utiliza o protocolo DNP3 para informar seus valores de leituras e sensores, procurou-se obter informações referentes ao tamanho máximo desses dados, chegando ao valor de 292 *bytes*.

Convém ressaltar que o local de armazenamento de dados (*buffer*) da interface serial onde o equipamento está conectado, está limitado em 255 *bytes*, necessitando assim, que seja alterado na biblioteca da placa, passando para 300 *bytes*.

Resolvida a questão da aquisição dos dados pela interface serial, agora faz-se necessário o envio destas informações recebidas do equipamento para o servidor de aplicação. Assim sendo, uma nova alteração na biblioteca se fez necessária devido a limitação em apenas 11 *bytes* para transmissão, sendo então alterado para 242 *bytes*, valor máximo quando utilizado o fator de espalhamento (SF) igual a 7, taxa de dados (DR) igual a 3 e banda de *uplink* de 125 kHz (kilo *Hertz*).

Realizadas as alterações, isso torna evidente que ainda não será possível enviar 292 *bytes* recebidos na interface serial em 242 *bytes* através da interface de rádio, fazendo-se necessária a criação de um protocolo interno para divisão dos pacotes. Desta forma, convencionou-se um cabeçalho com algumas informações, de modo a controlar e identificar o pacote, conforme visto no Quadro 15.

Quadro 15 – Transmissão módulo - Cabeçalho e dados

Campo	Tamanho	Intervalo/valores
Identificador de pacote	1 *byte*	0 até 100
Sequência do pacote	1 *byte*	de 1 até 2
Máximo de pacotes na sequência	1 *byte*	1 ou 2
Quantidade de *bytes*	1 *byte*	de 1 até 200
Dados	200 *bytes*	-

Fonte: Elaborado pelo autor.

O identificador de pacote: consiste em um contador para identificar que as informações enviadas em múltiplos pacotes pertencem a um mesmo conjunto de informações que foram divididas.

Sequência do pacote: informa que o pacote em questão é, por exemplo, o primeiro pacote de uma sequência de pacotes pertencentes a um mesmo conjunto de informações.

Máximo de pacotes na sequência: informa que um dado conjunto de informações foi dividido, por exemplo, em 3 pacotes para serem transmitidos de forma sequencial.

Quantidade de *bytes*: informa a quantidade de *bytes* que estão sendo transmitidos no pacote em questão.

Dados: são as informações propriamente ditas que se deseja encaminhar da interface serial para a interface de rádio, fechando assim, o ciclo de transmissão.

O cabeçalho convencionado ocupa 4 *bytes* da mensagem e, somados aos 200 *bytes* de dados, serão enviados, no máximo, 204 *bytes* em cada transmissão, ficando assim contido dentro do máximo permitido de 242 bytes e com a divisão em pacotes, tornando possível o envio dos 292 *bytes* do protocolo DNP3.

Vistas e implementadas as manipulações acima para criação do pacote, optou-se por aguardar o retorno de ACK proveniente do *gateway*, para que o pacote seguinte fosse enviado, respeitado o tempo entre envios de pacotes de 5000 ms (5 segundos).

No tópico seguinte, serão apresentadas as informações referentes aos testes de transmissão utilizando os dados do protocolo DNP3, de modo a garantir a integridade das informações desde a recepção na interface serial até o recebimento no servidor de aplicação.

4.4.3 Validação dos pacotes transmitidos com informações do protocolo DNP3

Após a implementação, realizou-se alguns testes triviais, para garantir a integridade das informações e os limites dos dados, de modo a verificar se todas as informações estavam sendo divididas e enviadas em cada pacote.

Inicialmente, verificou-se a integridade das informações e observou-se que, para alguns valores de *bytes* enviados, estes não chegavam de forma correta no servidor de aplicação. Como durante o desenvolvimento se utilizavam as sequências de letras maiúsculas/minúsculas, além dos números de 0 até 9, não se observou nenhuma anormalidade. Ao se utilizar valores em hexadecimal maiores ou iguais a 0x80, verificou-se que estes eram alterados, ficando sempre com valor igual a 0x3F. Este fato se deve a utilização da codificação de Base64, que é constituída apenas por 64 caracteres, sendo eles apresentados no Quadro 16.

Quadro 16 – Caracteres disponíveis na codificação Base64

	Caracteres	Hexadecimal	Decimal
Caracteres utilizados na codificação Base64	A até Z	0x00 até 0x19	0 até 25
	a até z	0x1A até 0x33	26 até 51
	0 até 9	0x34 até 0x3D	52 até 61
	+ e /	0x3E e 0x3F	62 e 63

Fonte: Elaborado pelo autor.

Com isso, comprova-se que não se trata de uma alteração das informações, mas sim de um limite na conversão da codificação Base64 para hexadecimal, que deve ocorrer no servidor de aplicação para que os dados retornem ao formato original. Identificado este problema, torna-se fundamental uma alteração no modo de envio.

Após algumas análises, resolveu-se pela segmentação dos bytes com valores em hexadecimal, transformando um *byte* em dois *bytes*, colocando os 4 *bits* da parte mais significativa em um *byte* e os 4 *bits* da parte menos significativa no *byte* seguinte, conforme demonstra o Quadro 17.

Quadro 17 – Segmentação de um *byte* em dois *bytes*

Dado recebido	0x01		0x02		0x81		0xC8	
Caractere (ASCII)	0	1	0	2	8	1	C	8
Dado transmitido	0x30	0x31	0x30	0x32	0x38	0x31	0x43	0x38

Fonte: Elaborado pelo autor.

Exposta a limitação em função da codificação Base64, a quantidade de *bytes* a serem transmitidos dobra, portanto, ao invés de realizar a transmissão de 292 *bytes*, serão 584 *bytes*, necessitando assim, também, alterar a quantidade de pacotes de 2 para 3, no caso extremo. Além disso, no servidor de aplicação, torna-se necessário o reagrupamento dos dois *bytes* em um *byte* novamente.

Após a realização das implementações necessárias, o problema não ocorreu e os dados transmitidos pelo módulo foram recebidos corretamente no servidor de aplicação.

Outro teste exploratório realizado teve como objetivo analisar os limites de dados em cada pacote, verificando se estavam sendo respeitados e enviados conforme pode-se observar no Quadro 18.

Quadro 18 – Análise dos limites de quantidade de *bytes* em cada pacote

Recebidos na serial	Enviados no rádio	Pacote 1	Pacote 2	Pacote 3
100 *bytes*	200 *bytes*	200 *bytes*	-	-
101 *bytes*	202 *bytes*	200 *bytes*	2 *bytes*	-
200 *bytes*	400 *bytes*	200 *bytes*	200 *bytes*	-
201 *bytes*	402 *bytes*	200 *bytes*	200 *bytes*	2 *bytes*
292 *bytes*	584 *bytes*	200 *bytes*	200 *bytes*	184 *bytes*
300 *bytes*	600 *bytes*	200 *bytes*	200 *bytes*	200 *bytes*

Fonte: Elaborado pelo autor.

No Quadro 19, são apresentadas as informações de depuração referentes a transmissão de 300 *bytes* (máximo possível implementado). Convém ressaltar que

esse valor é multiplicado por dois, devido a segmentação do *byte* recebido na serial, acrescido do cabeçalho de 4 *bytes*, sendo então, necessário o envio de 3 pacotes com 208 *bytes* em cada.

Quadro 19 – Informações de depuração, transmissão de um pacote de 300 *bytes*

Módulo transmissor	Servidor de Aplicação - MQTT
20:04:49.366 -> |id:2|seq:1|max:3|qtdBytesTX:208| 20:04:50.815 -> ACK RECEIVED	20:04:49,957 : messageArrived() added: message #1
20:04:53.769 -> |id:2|seq:2|max:3|qtdBytesTX:208| 20:04:55.232 -> ACK RECEIVED	20:04:54,360 : messageArrived() added: message #2
20:04:57.950 -> |id:2|seq:3|max:3|qtdBytesTX:208| 20:04:59.425 -> ACK RECEIVED	20:04:58,530 : messageArrived() added: message #3

Fonte: Elaborado pelo autor.

Pode-se observar um tempo médio de aproximadamente 587 ms, para transmitir cada pacote de 208 *bytes* entre o módulo e o servidor de aplicação. Já entre a transmissão do pacote e o recebimento do ACK, tem-se um tempo médio de latência de aproximadamente 1462 ms, para cada pacote.

Considerando as mesmas métricas adotadas no capítulo anterior, referentes ao cálculo da taxa de transferência, obtemos um valor aproximado de 354 B/s.

Um aspecto importante a ressaltar é o tempo entre as transmissões de dados. Esse intervalo foi alterado para valores menores que 5 segundos, mas não se obteve um bom resultado, pois em alguns casos os pacotes não chegavam ao seu destino ou então eram retransmitidos diversas vezes, o que aponta outra limitação na biblioteca, relacionada a temporizadores e lógica na máquina de estado de envio, recebimento de ACK e liberação para uma nova transmissão.

Tentou-se modificar a lógica de envios, de modo a não necessitar deste tempo. Assim, se utilizou como parâmetro “pronto para envio” a recepção do ACK, mas em algumas situações, verificou-se novamente os problemas de pacotes não chegarem ou então de demasiadas retransmissões de pacotes. Com isto, se optou por manter o tempo de guarda de 5 segundos, onde não foram verificados problemas em inúmeras transmissões de testes realizadas durante a validação desta implementação.

Com isto, encerra-se a implementação e a validação da transmissão de dados, realizada pelo módulo, e a recepção pelo servidor de aplicação. Assim, no tópico a seguir, será tratada a parte de transmissão de dados pelo servidor de aplicação e recepção de dados pelo módulo.

4.4.3 Recepção de pacotes com informações do protocolo DNP3

Viu-se, inicialmente, que a biblioteca fornecida pelo fabricante realiza a recepção de dados, conforme demonstrado no início deste capítulo, porém, o modo é muito primitivo, sem tratamento de inicialização e controle das variáveis responsáveis pela recepção dos dados. Com isto, constatou-se a necessidade de alteração na biblioteca.

Inicialmente, foi identificada a utilização de uma mesma variável para realização da transmissão e da recepção de dados, visto ainda durante testes iniciais, isto demonstrou-se como um grande problema, fazendo-se necessário a criação de uma variável para cada sentido da comunicação.

Verificou-se que, ao receber uma grande quantidade de dados, o *buffer* de recepção não era limpo após o tratamento e, portanto, em uma recepção seguinte, haveria informações misturadas. Para evitar isso, foram tomadas medidas após o tratamento das informações recebidas, de modo a realizar a limpeza das variáveis de *buffer*, a quantidade de dados recebidos e o controle de novos dados.

Após as alterações expostas, passou-se então para o desenvolvimento propriamente dito, que envolve o tratamento das informações recebidas do servidor de aplicação na interface de rádio, para posterior envio para a interface serial.

Toda a infraestrutura desenvolvida para a transmissão de dados e para a segmentação de um *byte* em dois, é necessária para o tratamento dos dados recebidos, diferenciando-se pelo fato de agora receber já dividido em dois *bytes*, tendo que reagrupá-los novamente em um *byte*, conforme pode ser visto no Quadro 20.

Quadro 20 – Reagrupamento de dois *bytes* em um *byte*

Dados recebidos	0x30	0x31	0x30	0x32	0x38	0x31	0x43	0x38
Caractere (ASCII)	0	1	0	2	8	1	C	8
Dados transmitidos	0x01		0x02		0x81		0xC8	

Fonte: Elaborado pelo autor.

Realizada a reconstrução do *byte*, faz-se necessário avaliar se o pacote recebido faz parte de uma sequência de pacotes ou, então, se é único, conforme verificação realizada nos 4 primeiros *bytes* do pacote recebido, que se referem ao cabeçalho, conforme apresentado no Quadro 21.

Quadro 21 – Recepção do módulo - Cabeçalho e dados

Campo	Tamanho	Intervalo/valores
Identificador de pacote	1 *byte*	0 até 100
Sequência do pacote	1 *byte*	de 1 até 3
Máximo de pacotes na sequência	1 *byte*	1, 2 ou 3
Quantidade de *bytes*	1 *byte*	de 1 até 200
Dados	200 *bytes*	Base64

Fonte: Elaborado pelo autor.

Se o pacote for de uma sequência, é necessário aguardar os demais para que a informação do protocolo DNP3 seja enviada em sua totalidade, uma vez que a mesma está ligada ao equipamento que receberá estas informações, para que ações sejam tomadas.

O processo de validação, no caso da recepção, foi executado juntamente com o desenvolvimento, visto que muitas limitações impostas pela biblioteca tiveram que ser contornadas, partindo-se, assim, para novos testes.

Durante a realização de testes onde o servidor de aplicação enviava informações para o módulo LoRa, verificou-se que pacotes com mais de 30 *bytes* não chegavam até o módulo. Ao se analisar este comportamento, observou-se que estes não estavam chegando ao *gateway* e, com isso, temos um grande impedimento, no qual muito tempo seria empregado para analisar e contornar o problema.

Convém ressaltar que, em virtude dos equipamentos *gateway* e servidor de aplicação não fazerem parte direta do presente trabalho, optou-se por admitir, por convenção, que o recebimento e o tratamento de pacotes aconteceriam de modo adequado, caso não houvessem limitações no servidor de aplicação.

No tópico seguinte, algumas análises serão realizadas, referentes a transmissão de dados do protocolo DNP3 e, dadas as limitações de recepção, essas serão na medida do possível verificadas com comandos que não ultrapassem os 30 *bytes* de limitação.

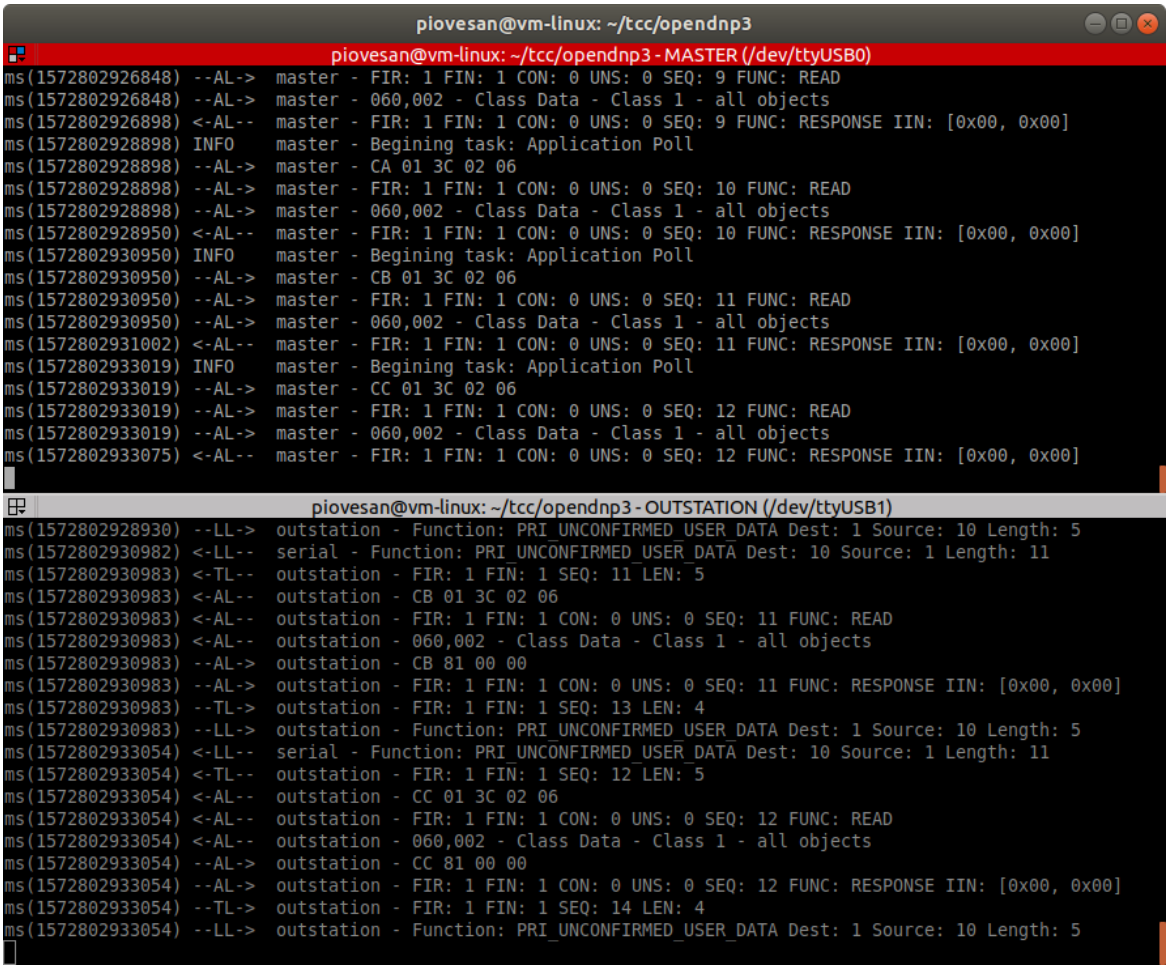
4.5 Avaliação do protocolo DNP3 sobre o LoRaWAN

Devido às limitações físicas da utilização direta de um equipamento que disponibiliza suas informações utilizando o protocolo DNP3, optou-se inicialmente por

utilizar um simulador do protocolo DNP3, que simula o protocolo realizando o papel de mestre (*Master*) e escravo (*Outstation*).

Este simulador está disponibilizado em (GITHUB DNP3, 2019). Entretanto, necessitou-se realizar modificações para que a interface serial pudesse ser utilizada, mas limitando-se apenas a esta alteração, sendo seu funcionamento observado na Figura 31.

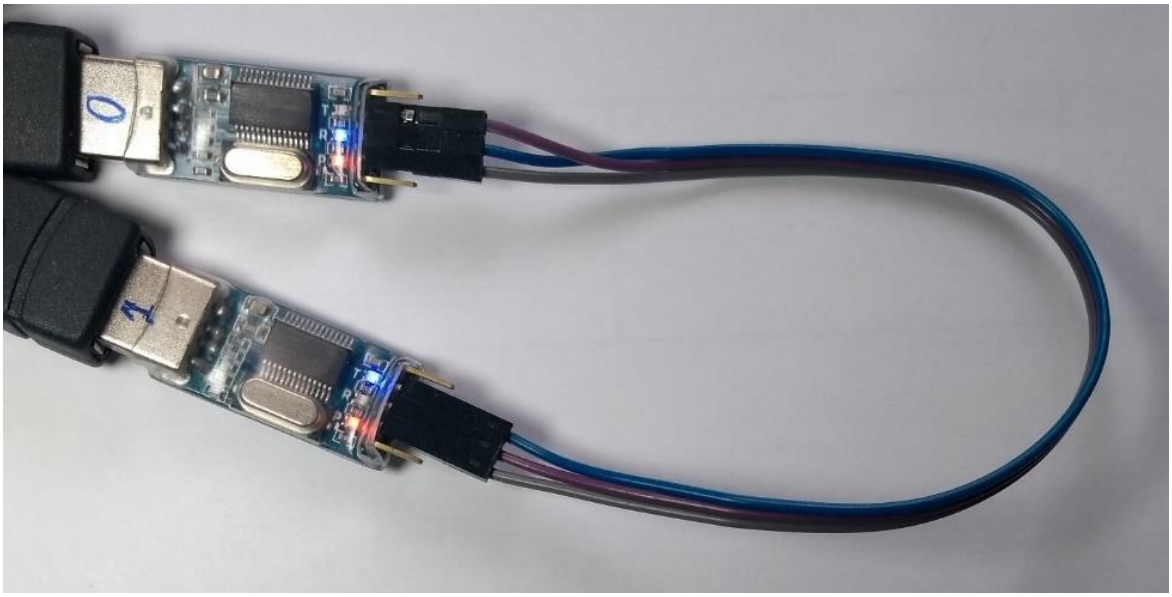
Figura 31 – Comunicação *Master-Outstation* do simulador



Fonte: Elaborado pelo autor.

Para que o teste acima pudesse ser realizado, necessitou-se de conversores de interface serial para interligação dos mesmos e, para isso, na parte superior da Figura 31, pode-se observar os dados sendo enviados pelo *Master,* utilizando a interface serial do *Linux* "*/dev/ttyUSB0*", enquanto que, na parte inferior da Figura 31, observa-se o *Outstation* recebendo as requisições do *Master*, na interface serial do *Linux* "*/dev/ttyUSB1*". *Estas* interfaces são físicas e conectadas ao computador, conforme observa-se na Figura 32.

Figura 32 – Conexão serial teste inicial do simulador de protocolo DNP3



Fonte: Elaborado pelo autor.

Com o teste acima realizado com sucesso, pode-se, então, realizar a captura de comandos do protocolo DNP3 enviados pelo *Master*, para realização de testes manuais, sendo alguns exemplos destes comandos apresentados no Quadro 22.

Quadro 22 – Exemplo de comandos protocolo DNP3

Comandos do protocolo DNP3 – *Master* (formato hexadecimal)	Quantidade de *bytes*
05 64 11 C4 0A 00 01 00 06 15 C0 C0 15 3C 02 06 3C 03 06 3C 04 06 1A 55	24
05 64 05 C9 0A 00 01 00 FE DA	10

Fonte: Elaborado pelo autor.

Convém ressaltar que existe a limitação de 30 *bytes* no sentido de transmissão do servidor de aplicação para o módulo. Portanto, optou-se pela utilização do comando com 10 *bytes*, sendo este o de menor quantidade, pois como visto anteriormente, devem ser adicionados 4 *bytes* de cabeçalho, totalizando 14 *bytes*. Durante a conversão para a transmissão, este dobrará de tamanho, ficando com 28 *bytes*.

No tópico seguinte, será utilizado o comando de 10 *bytes* para a realização da comunicação entre o simulador do *Master* (servidor de aplicação) e o simulador do *Outstation* (módulo), de modo a exercitar um ciclo da comunicação.

4.5.1 Transmissão de comandos DNP3 do servidor de aplicação para módulo

De posse do comando, os testes no sentido de comunicação do servidor de aplicação para o módulo, se deram do seguinte modo:

A partir do servidor de aplicação, enviou-se o cabeçalho, juntamente com o comando hexadecimal, conforme apresentado no Quadro 23.

Quadro 23 – Representação dos dados enviados servidor aplicação hexadecimal

Cabeçalho	Comando
00 01 01 14	05 64 05 C9 0A 00 01 00 FE DA

Fonte: Elaborado pelo autor.

No Quadro 24, é apresentada a representação do cabeçalho e comando em Base64, pois é deste modo que o envio é realizado pelo servidor de aplicação.

Quadro 24 – Representação dos dados enviados servidor aplicação em Base64

Base64	MDAwMTAxMTQwNTY0MDVDOTBBMDAwMTAwRkVEQQ==

Fonte: Elaborado pelo autor.

No módulo, foram recebidas as informações acima e, após reverter a codificação Base64, encontrou-se novamente os dados originalmente enviados, conforme pode ser observado nas mensagens de depuração, no Quadro 25.

Quadro 25 – Dados recebidos no módulo

```
10:38:42.087 -> | bufferRadioRX: 0x00 | HIGH: 0x00 | LOW: 0x0 |
10:38:42.189 -> | bufferRadioRX: 0x01 | HIGH: 0x00 | LOW: 0x1 |
10:38:42.325 -> | bufferRadioRX: 0x01 | HIGH: 0x00 | LOW: 0x1 |
10:38:42.427 -> | bufferRadioRX: 0x14 | HIGH: 0x10 | LOW: 0x4 |
10:38:42.529 -> | bufferRadioRX: 0x05 | HIGH: 0x00 | LOW: 0x5 |
10:38:42.631 -> | bufferRadioRX: 0x64 | HIGH: 0x60 | LOW: 0x4 |
10:38:42.733 -> | bufferRadioRX: 0x05 | HIGH: 0x00 | LOW: 0x5 |
10:38:42.835 -> | bufferRadioRX: 0xc9 | HIGH: 0xc0 | LOW: 0x9 |
10:38:42.937 -> | bufferRadioRX: 0x0a | HIGH: 0x00 | LOW: 0xa |
10:38:43.039 -> | bufferRadioRX: 0x00 | HIGH: 0x00 | LOW: 0x0 |
10:38:43.175 -> | bufferRadioRX: 0x01 | HIGH: 0x00 | LOW: 0x1 |
10:38:43.277 -> | bufferRadioRX: 0x00 | HIGH: 0x00 | LOW: 0x0 |
10:38:43.379 -> | bufferRadioRX: 0xfe | HIGH: 0xf0 | LOW: 0xe |
10:38:43.481 -> | bufferRadioRX: 0xda | HIGH: 0xd0 | LOW: 0xa |
10:38:43.483 -> | COMMAND SEND TO SERIAL: 05 64 05 C9 0A 00 01 00 FE DA |
```

Fonte: Elaborado pelo autor.

Deste modo, a informação originalmente enviada pelo servidor de aplicação consegue encerrar seu ciclo, chegando a serial do módulo, que está conectada ao simulador de protocolo DNP3, que processará o comando recebido, respondendo novamente para o módulo e iniciando o ciclo de transmissão do módulo para o servidor de aplicação, o qual será visto no tópico a seguir.

4.5.2 Transmissão de comandos DNP3 do módulo para servidor de aplicação

Ao receber o comando DNP3, o simulador de *Outstation* verifica as informações solicitadas, gerando um novo comando de resposta para o simulador *Master*, conforme pode ser visto no Quadro 26, juntamente com a informação de cabeçalho adicionada pelo módulo.

Quadro 26 – Representação dos dados enviados pelo módulo em hexadecimal

Cabeçalho	Comando
00 01 01 14	05 64 05 0B 01 00 0A 00 6D ED

Fonte: Elaborado pelo autor.

No Quadro 27, é apresentada a representação do cabeçalho e comando em Base64, pois é deste modo que o servidor de aplicação receberá a informação.

Quadro 27 – Representação dos dados enviados pelo módulo em Base64

Base64	MDAwMTAxMTQwNTY0MDUwQjAxMDAwQTAwNkRFRA==

Fonte: Elaborado pelo autor.

Ao receber a informação mostrada no Quadro 27, esta é repassada ao simulador de *Master*, concluindo assim, o ciclo de comunicação entre *Master* e *Outstation* utilizando o protocolo DNP3, conforme pode ser visto, em detalhes, ao longo do tempo no tópico a seguir.

4.5.3 Ciclo completo de recepção e transmissão de comandos DNP3

Este tópico destina-se a avaliar o tempo desde o momento em que o comando é enviado do simulador *Master*, até que este receba novamente a mensagem de resposta, proveniente do simulador *Outstation*.

Para isso, serão considerados valores máximos suportados para os pacotes, ou seja, no sentido de transmissão do servidor de aplicação para o módulo, utiliza-se 30 *bytes* (devido às limitações do servidor de aplicação, já vistas) e, no sentido de transmissão do módulo para o servidor de aplicação, serão necessários 2 pacotes com 200 *bytes* cada e outro de 184 *bytes*, totalizando 584 *bytes*, além de 8 *bytes* de cabeçalho em cada um dos 3 pacotes. Deste modo, será possível avaliar o tempo de *timeout* necessário ao simulador *Master*, até que este receba a resposta, considerando, para isso, o pior caso.

No Quadro 28, são apresentadas as mensagens de depuração de um ciclo completo de comunicação, que se inicia pelo servidor de aplicação (simulador *Master*), passando pelo simulador *Outstation*, que está conectado a interface serial do módulo e se encerrando ao retornar ao servidor de aplicação.

Quadro 28 – Ciclo completo de comunicação utilizando o protocolo DNP3

Informações de depuração do servidor de aplicação juntamente com as do módulo
17:30:00,625 : sucessfully published message {
17:30:03.108 -> ### DEBUG | SEND COMMAND TO SERIAL INTERFACE
17:30:05.213 -> ### DEBUG | RECEIVED COMMAND ON SERIAL INTERFACE
17:30:05.654 -> ### DEBUG | SENT PACKET 1/3 (data: 208)(header: 8)
17:30:06,189 : messageArrived() added: message #1
17:30:06.979 -> ### DEBUG | ACK RECEIVED (1/3)
17:30:11.733 -> ### DEBUG | SENT PACKET 2/3 (data: 208)(header: 8)
17:30:12,274 : messageArrived() added: message #2
17:30:13.057 -> ### DEBUG | ACK RECEIVED (2/3)
17:30:17.679 -> ### DEBUG | SENT PACKET 3/3 (data: 184)(header: 8)
17:30:18,226 : messageArrived() added: message #3
17:30:19.005 -> ### DEBUG | ACK RECEIVED (3/3)
17:30:23.490 -> ### STATE (2) DEVICE READY TO SEND DATA AGAIN ###

Fonte: Elaborado pelo autor.

Nas informações de depuração apresentadas no Quadro 28, pode-se observar que o tempo entre a transmissão do comando e a recepção da resposta deste comando no simulador operando como *Master* é de 17601 ms ou, aproximadamente, 17 segundos, considerando-se o momento inicial quando o comando foi publicado e o final, quando o servidor de aplicação receber a terceira parte do pacote.

Conforme dito anteriormente, houve tentativas de alterar o comportamento entre os envios de pacotes, de modo a não aguardar cerca de 5 segundos, mas houve grandes problemas, tais como perda de pacotes e retransmissões, visto que o ACK não era recebido em alguns casos.

Pelo exposto acima, recomenda-se a utilização, no *Master*, de um tempo mínimo (*timeout*) de 20 segundos entre envio e resposta deste comando, visto que se trata de uma extrapolação utilizando-se o maior pacote possível no protocolo DNP3, que contêm 292 *bytes*.

4.5.4 Considerações a respeito da forma do teste realizado

Dadas as limitações físicas no quesito de acesso ao equipamento que se comunica utilizando o protocolo DNP3, optou-se, conforme falado anteriormente, na utilização dos simuladores para *Master* e *Outstation*, salvo pela outra limitação imposta pelo servidor de aplicação, que não envia pacotes maiores que 30 *bytes*. Dada a existência de um módulo ou *software* que fizesse a convergência das informações do *Master* para o servidor de aplicação, o trabalho elaborado poderia ser testado de modo eficaz utilizando um cenário real.

Deste modo e vistas essas considerações, sugere-se futuramente trabalhar com a interligação real dos equipamentos. Convém ressaltar alguns aspectos relacionados ao tempo (*timeout*) que os equipamentos ou simuladores do protocolo DNP3 aguardam após o envio de um comando até o recebimento de uma resposta.

No simulador do protocolo DNP3 utilizado por padrão, utiliza-se um *timeout* até o recebimento de 20 segundos. Portanto, os 17 segundos encontrados no teste do tópico anterior, atenderiam. Faz necessário ressaltar que os testes foram feitos em ambiente com apenas a comunicação de um módulo, sendo assim, não há como prever a influência de outros módulos comunicando simultaneamente com o *gateway*.

Uma vez que foi trabalhado muito próximo do seu limiar de atuação, deve-se considerar muito bem a utilização de módulos LoRa em redes que sejam de grande criticidade e quantidade de informações, o que não é uma imposição de limitação, mas sim um alerta, visto que constatamos a eficácia em transmissões de longas distâncias.

5 CONSIDERAÇÕES FINAIS

Muitos desafios foram encontrados ao longo do desenvolvimento do presente trabalho e, de uma maneira ou outra, boa parte deles foram contornados, dado que o módulo utilizado conta ainda com constantes atualizações em sua biblioteca, uma vez que não é muito usual sua operação em classe C, visto que a mesma não preza pela economia de energia quando comparada com as classes A e B.

Em alguns momentos as limitações levaram a crer que o objetivo de utilizar o módulo LoRa como meio de acesso à interface serial de equipamentos que utilizam o protocolo DNP3 pudesse não ser atingido, o que demandou mais esforço para analisar as causas dos problemas e corrigi-las de forma adequada.

De acordo com os testes de taxa de transferência, latência e grandes quantidades de dados, observou-se que existem limitações quanto à banda e tamanho de pacotes, as quais são conhecidas e explicitadas na especificação da tecnologia LoRa e do protocolo LoRaWAN, mas as mesmas não configuram um impeditivo para a utilização, uma vez que que é possível utilizar-se de artifícios, tais como a divisão em pacotes e/ou diminuição do tempo das janelas de transmissão.

Com isso, é necessário ressaltar que, dado o modo de operação do protocolo DNP3 não estar condicionado a apenas responder à requisições, o equipamento *Outstation*, através de suas configurações pode, a qualquer momento, enviar informações sem que um *Master* as tenha requisitado. Desta forma, torna-se difícil o gerenciamento das informações no meio transmissão que, neste caso, é representado pela rede LoRa utilizando-se do protocolo LoRaWAN para transporte das informações geradas pelos equipamentos.

Esta questão não torna inviável o uso de rádios LoRa como meio de transmissão, conforme visto no final do capítulo anterior, mas se coloca como item a ser analisado com bastante prudência, pois podem ocorrer perdas de informação ou até mesmo colisão de dados na interface serial do módulo. Como meio de contorno, sugere-se embarcar o rádio LoRa diretamente dentro dos equipamentos, sendo assim controlado pelo mesmo microcontrolador ou processador que gerencia todas as tarefas no equipamento. Deste modo, o rádio LoRa opera como uma interface de acesso RS-232, RS-485 (*Recommended Standard 485*) ou *ethernet*, normalmente disponibilizadas em equipamentos que necessitam de gerenciamento remoto ou local.

As aplicações de sensores que utilizam rádios LoRa para comunicação, normalmente operam utilizando as classes A e B, onde dispõem de janelas específicas para recepção de dados. Portanto, torna-se trivial saber os momentos em que não podem ser realizadas transmissões para se evitar colisões ou outros efeitos indesejados, tais como a perda de pacotes.

Embora neste trabalho tenham sido adotadas métricas para evitar a transmissão de pacotes em momentos de recepção ou vice-versa, isso não evita que tanto o equipamento *Master* quanto o *Outstation* gerem informações, sendo que estas podem vir a ser perdidas uma vez que se optou pela priorização de tarefas previamente iniciadas.

Os métodos de validação da integridade dos dados também foram extremamente importantes, pois possibilitaram, ainda durante o desenvolvimento do código a ser embarcado no módulo, a identificação de problemas e limitações, visto que tal prática é comum em empresas de desenvolvimento de *software*, pois permite garantir entregas com qualidade, por mais que muitas pessoas ainda contestem os resultados, afirmando que agregam tempo no desenvolvimento.

Durante a realização do teste de longa distância, identificou-se que, na topologia ponto a ponto, a taxa de transferência era um pouco menor que a especificada nas documentações do protocolo. Considerando a distância de aproximadamente 3,4 km que o módulo transmissor se encontrava do receptor e, visto estar operando no limiar de sua recepção, podemos justificar tal comportamento. Já em testes com a topologia utilizando-se do *gateway*, obteve-se um desempenho dentro da especificação do protocolo LoRaWAN, mesmo estando em um ambiente com maior probabilidade de interferências eletromagnéticas.

Outro ponto importante a destacar está relacionado ao consumo de energia. Atualmente se, quando o assunto é a Internet das Coisas, fala-se serem necessários módulos com baixo consumo para inúmeras aplicações. Durante a realização dos testes de campo, utilizou-se acumuladores de energia (*Power Bank*), normalmente utilizados para cargas de emergência em celulares, sendo que estes proveram autonomia por aproximadamente 20 horas, mesmo sendo realizadas transmissões a cada segundo e com a utilização de *display* de OLED, o que demonstrou ser uma excelente opção para situações que necessitem de alimentação através de baterias.

Como trabalhos futuros, temos as seguintes sugestões:

A agregação do rádio em conjunto com microcontroladores/microprocessadores, fazendo a divulgação de variáveis de leituras analógicas e digitais, assim como o recebimento de informações para acionamentos, utilizando-se de qualquer uma das classes disponíveis para a modulação LoRa.

O desenvolvimento de experimentos para montagem de antenas com grandes ganhos, a fim de melhorar o alcance e utilizando-se de módulos de rádio com potência de 1 W, como muitos modelos disponíveis no mercado.

A exploração de outras classes de operação, visando estimar a durabilidade da bateria que energiza o módulo de rádio e sensores, para aplicações onde se faça necessária tal utilização, como em ambientes remotos e de difícil acesso.

O desenvolvimento de uma rede com inúmeros módulos conectados a um *gateway*, de modo a validar questões de disponibilidade e confiabilidade no recebimento das informações geradas pelos módulos LoRa.

E, por fim, a transmissão de imagens em baixa qualidade, através do rádio LoRa, em conjunto com o protocolo LoRaWAN, utilizando-se de câmeras ligadas diretamente a microcontroladores/microprocessadores, de modo a informar sobre movimentos.

REFERÊNCIAS

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS (ABNT). **ABNT NBR 27002**: tecnologia da informação – técnicas de segurança – código de prática para a gestão da segurança da informação. Rio de Janeiro: ABNT, 2005.

CLARKE, Gordon; REYNDERS, Deon; WRIGHT, Edwin. **Practical Modern SCADA Protocols: DNP3, 60870.5 and Related Systems**. 1. Ed. Newnes, 2004. *E-book*.

GITHUB DNP3; **dnp3/opendnp3**. Automatak LLC, [2019?]. Disponível em: https://github.com/dnp3/opendnp3. Acesso em: 8 outubro 2019. SHA, 44c06cb148d40ed150e539895b3fa951376e5f74.

GITHUB HELTEC; **HelTecAutomation/ESP32_LoRaWAN**. HELTEC, [2019?]. Disponível em: https://github.com/HelTecAutomation/ESP32_LoRaWAN. Acesso em: 16 setembro 2019. SHA, 501f6d82ee9fc8f6b431fbdb2b3913643b50b495.

HELTEC. **WiFi LoRa 32 (V2)**. HELTEC, [2019?]. Disponível em: https://heltec.org/project/wifi-lora-32/. Acesso em: 28 maio 2019.

LORA ALLIANCE. **What is the LoRaWAN® Specification?.** LORA ALLIANCE, [2019?]. Disponível em: https://lora-alliance.org/about-lorawan. Acesso em: 19 maio 2019.

MACKAY, Steve; WRIGHT, Edwin; REYNDERS, Deon; PARK, Jonh**. Practical Industrial Data Networks: Design, Installation and Troubleshooting**. 1. Ed. Newnes, 2004. *E-book*.

MINERVA, Roberto; BIRU, Abyi; ROTONDI, Domenico. **Towards a definition of the Internet of Things (IoT)**. IEEE, 2015. Disponível em: https://iot.ieee.org/images/files/pdf/IEEE_IoT_Towards_Definition_Internet_of_Things_Revision1_27MAY15.pdf. Acesso em: 4 abr. 2019.

PRADO, Edmir; SOUZA, Cesar Alexandre de. **Fundamentos de sistemas de informação**. 1. Ed. Rio de Janeiro: Campus (Elsevier), 2014. *E-book* (não paginado). Disponível em: https://www.evolution.com.br/epubreader/9788535274363. Acesso em: 27 mar. 2019.

SEMTECH. **What is LoRa®?**. SEMTECH, [2019?]. Disponível em: https://www.semtech.com/lora/what-is-lora. Acesso em 5 maio 2019.

SENEVIRATNE, Pradeeka**. Beginning LoRa Radio Networks with Arduino: Build Long Range, Low Power Wireless IoT Networks**. 1. Ed. Apress, 2019. *E-book*.

SPARKFUN. **RS-232 vs. TTL Serial Communication**. SparkFun, [2019?]. Disponível em: https://www.sparkfun.com/tutorials/215. Acesso em:3 jun. 2019.

TANENBAUM, Andrew S. **Redes de computadores**. 4. ed. Rio de Janeiro: Campus (Elsevier), 2003.

TI. **Spectrum Analyzer (MSP-SA430-SUB1GHZ)**. TEXAS INSTRUMENTS, [2019?]. Disponível em: https://www.ti.com/store/ti/en/p/product/?p=MSP-SA430-SUB1GHZ. Acesso em: 31 maio 2019.

XIAO, Perry. **Designing Embedded Systems and the Internet of Things (IoT) with the ARM® Mbed™**. 1. Ed. Hoboken: Wiley, 2018. *E-book*.