

Modelo protótipo de sistema *smart grid* com rede LPWAN para comunicação

Mestrado em Engenharia Eletrotécnica e de Computadores

Vinicius Pedroza Delsin

Orientação
Ramiro Barbosa
Antonio Newton Licciardi Junior

Ano Letivo: 2021-2022

Instituto Superior de Engenharia do Porto
Departamento de Engenharia Eletrotécnica
Rua Dr. António Bernardino de Almeida, 431, 4200-072 Porto

Resumo

Os avanços em eletrônica digital e sistemas de telecomunicação possibilitaram a integração de uma cadeia de fluxo de informações em diversos setores da sociedade, para as redes elétricas não é diferente.

Este trabalho tem como objetivo construir e detalhar um modelo protótipo de um sistema de *smart grid* completo utilizando rádio LoRa como forma de comunicação. Para isso, este protótipo visa percorrer toda a cadeia de informação iniciada no *smart meter* até a visualização dos dados coletados. Por fim, este trabalho visa apresentar a utilização de um modelo de inteligência artificial para a previsão de demanda do consumidor.

Palavras-chave: Rede Elétrica Inteligente, Medidor Inteligente, Sistema Elétrico de Potência, *Smart Grid*, *Smart Meter*, LoRa.

Abstract

Advances in digital electronics and telecommunication systems made it possible to integrate a chain of information flow in several sectors of society, in particular the electrical grid, which enabled the exchange of information between concessionaires and consumers.

This work aims to build and detail a prototype model of a complete smart grid system using LoRa radio for communication. For this, this prototype aims to go through the entire information chain started in the smart meter until the visualization of the collected data. Finally, this work aims to present the use of an artificial intelligence model to forecast consumer demand.

Keywords: Electric Power Systems, *Smart Grid*, *Smart Meter*, LoRa.

Conteúdo

Conteúdo	v
Lista de Figuras	ix
Lista de Tabelas	xi
Lista de Siglas e Abreviaturas	xiii
Agradecimentos	xv
1 Introdução	1
1.1 Contexto	1
1.2 Justificativa	4
1.3 Objetivos	6
1.4 Calendarização	7
1.5 Estrutura	8
2 Estado da arte	9
2.1 <i>Smart Grid</i>	9
2.1.1 Cenário Atual dos Sistemas de Rede Elétrica	9
2.1.2 <i>Smart Grid</i> ou Redes Inteligentes	10
2.1.3 <i>Microgrids</i>	12
2.1.4 Experiências de Implementação	13
2.1.5 Desafios	14
2.1.6 Perspectiva para o Futuro	14
2.2 <i>Smart Meter</i>	15
2.2.1 <i>Smart Meter</i> ou Medidor Inteligente	15
2.2.2 Características e Funcionalidades	15
2.3 LPWAN, <i>Low Power Wide Area Network</i>	16
2.3.1 Conceitos de modulação	16
2.3.2 Redes LPWAN	19

2.3.3	LoRa e LoRaWAN	20
2.4	MQTT - <i>Message Queuing Telemetry Transport</i>	22
2.4.1	Protocolo MQTT	22
2.4.2	Características e Formas de Comunicação	22
2.4.3	Eclipse Mosquitto	24
2.5	Modelos de Previsão para Demanda Energética	24
2.5.1	Demandá Energética e Curva de Carga	24
2.5.2	Inteligência Artificial	25
3	Hardware	27
3.1	Esquemático do Sistema	27
3.2	<i>Smart Meter</i>	28
3.2.1	Esquemático do <i>Smart Meter</i>	28
3.2.2	Sensor Tensão - ZMPT101B	29
3.2.3	Sensor Corrente - ACS712T	30
3.2.4	Microcontrolador <i>Smart Meter</i> - ESP32	32
3.2.5	Rádio LoRa <i>Smart Meter</i> - 433T20D	33
3.3	<i>Gateway</i>	34
3.3.1	Esquemático do <i>Gateway</i>	34
3.3.2	Microcontrolador <i>Gateway</i> - ESP32	35
3.3.3	Rádio LoRa <i>Gateway</i> - 433T20D	35
4	Software	37
4.1	Código <i>Smart Meter</i>	37
4.1.1	Fluxograma - Cálculo da Tensão	37
4.1.2	Fluxograma - Cálculo da Corrente	39
4.1.3	Fluxograma - Cálculo da Frequência e Envio de Dados LoRa	40
4.2	Código <i>Gateway</i>	42
4.2.1	Fluxograma - Conexão Wifi e Recepção dos Dados via LoRa	42
4.3	Código MQTT	43
4.3.1	Descrição - Conexão com o Servidor MQTT	43
4.4	Código Banco de Dados	44
4.4.1	Descrição - Conexão com o Servidor de Banco de Dados .	44
4.5	Ligaçāo com <i>Dashboard</i>	45
4.5.1	Descrição - Conexão entre Banco de Dados e <i>Dashboard</i> .	45
4.6	Descrição - Algoritmo de Modelo de Previsão de Carga	46
4.6.1	Tratamento do Banco de Dados	46
4.6.2	Algoritmo de Previsão de Carga	48
5	Resultados e Discussões	51
5.1	Testes de Precisão e Confiabilidade do Sistema	51
5.1.1	Teste de Precisão do ZMPT101B	51

5.1.2	Teste de Precisão do ACS712T	53
5.2	Funcionamento Completo do Sistema <i>Smart Grid</i>	54
5.2.1	Funcionamento do Sistema <i>Smart Grid</i>	54
5.2.2	Aquisição dos Dados por parte do <i>Smart Meter</i>	55
5.2.3	Transmissão dos Dados do <i>Smart Meter</i> e Recepção dos Dados pelo <i>Gateway</i> via Rádio LoRa	55
5.2.4	Encaminhamento dos Dados Recebidos pelo <i>Gateway</i> para um Servidor MQTT via Wifi	56
5.2.5	Armazenamento dos Dados Recebidos pelo Servidor MQTT em um Banco SQL	57
5.2.6	Monitoramento dos Dados Armazenados no Banco SQL através de uma Visualização Gráfica	58
5.2.7	Análise Estatística dos Dados através de Algoritmos de Inteligência Artificial	59
6	Conclusões	61
6.1	Considerações Finais	61
6.1.1	Objetivos Alcançados	61
6.1.2	Dificuldades	61
6.1.3	Aplicações Práticas e Trabalhos Futuros	62
Referências		63

Lista de Figuras

2.1	Topologia de um Sistema Elétrico de Potência [1].	10
2.2	Topologia de um Sistema <i>Smart Grid</i> [2].	11
2.3	Exemplo de <i>Microgrids</i> [3].	13
2.4	Exemplo de <i>Smart Meter</i> [4].	15
2.5	Exemplo de Modulação de Frequência [5].	17
2.6	Exemplo de Modulação de Amplitude [5].	18
2.7	Exemplo de Modulação PSK, FSK e ASK [5].	19
2.8	Janela de Recepção de Dispositivos Classe A [6].	21
2.9	Janela de Recepção de Dispositivos Classe B [6].	21
2.10	Janela de Recepção de Dispositivos Classe C [6].	22
2.11	Exemplo do protocolo MQTT [7].	23
2.12	Curva de Carga Residencial do Brasil entre 2008 e 2021 [8].	24
2.13	Curva de Carga Mensal Residencial do Brasil em 2021 [8].	25
2.14	Curva de Carga Horária Típica do Sistema Elétrico [9].	25
3.1	Esquemático do Sistema.	27
3.2	Esquemático do <i>Smart Meter</i>	28
3.3	Imagen do Protótipo <i>Smart Meter</i>	28
3.4	Sensor de Tensão ZMPT101B [10].	29
3.5	<i>Pinout</i> do Sensor de Tensão ZMPT101B [10].	29
3.6	Sensor de Corrente ACS712T.	30
3.7	Tabela de Precisão do Sensor ACS712T [11].	31
3.8	<i>Pinout</i> do Sensor ACS712T [12].	31
3.9	Placa ESP32 Devkit v1.	32
3.10	<i>Pinout</i> da Placa de Desenvolvimento ESP32 Devkit v1 [13].	32
3.11	Placa LoRa E32-433T20D.	33
3.12	<i>Pinout</i> da Placa LoRa E32-433T20D [14].	33
3.13	Esquemático do <i>Gateway</i>	34
3.14	Imagen do Protótipo <i>Gateway</i>	34

4.1	Fluxograma Código do Cálculo de Tensão do <i>Smart Meter</i>	38
4.2	Fluxograma Código do Cálculo da Corrente do <i>Smart Meter</i>	39
4.3	Fluxograma Código do Cálculo da Frequência e Envio de Dados LoRa do <i>Smart Meter</i>	41
4.4	Fluxograma Código Conexão Wifi e Recepção via LoRa do <i>Gateway</i> . .	42
4.5	<i>Loop</i> de Requisição ao Servidor Web Roteado pelo <i>Gateway</i>	43
4.6	Adição de Data/Hora e Envio do Pacote ao Servidor MQTT.	44
4.7	<i>Loop</i> de Funcionamento e Parâmetros de Conexão.	44
4.8	Função de <i>Callback</i> para Publicações no Tópico MQTT.	45
4.9	Função de Armazenamento de Dados no Banco de Dados.	45
4.10	Código de Consulta ao Banco de Dados.	46
4.11	Código para Visualização dos Dados	46
4.12	Código para Definir o Tipo de Consumo.	47
4.13	Código para Padronizar as Datas.	47
4.14	Código para Eliminar Colunas e Converter para Extensão csv. . . .	47
4.15	Código para Separação dos Dados entre Treino e Teste.	48
4.16	Código para Transformação dos Dados.	48
4.17	Código para Definição dos <i>Inputs</i> e <i>Outputs</i> do Algoritmo.	49
4.18	Código para Definição do Algoritmo LSTM.	49
4.19	Código para Transformação Inversa dos Dados.	49
4.20	Código para Construção da Tabela e Visualização Comparativa. . .	49
4.21	Código para Implementação da Coluna de MAPE.	50
5.1	Teste de Confiabilidade do ZMPT101B com Alimentação de 5,0 V. . .	52
5.2	Teste de Confiabilidade do ZMPT101B com Alimentação de 3,3 V. . .	52
5.3	Conversor analógico-digital do microcontrolador ESP32.	53
5.4	Teste de Confiabilidade do ACS712T.	54
5.5	<i>Log</i> dos Dados através do Monitor Série.	55
5.6	Código Responsável pela Transmissão dos Dados.	55
5.7	Código Responsável pela Recepção dos Dados.	56
5.8	Exemplo de Conexão Wifi do <i>Gateway</i>	56
5.9	Código Responsável pelo Estabelecimento do Servidor <i>web</i>	56
5.10	<i>Log</i> de Mensagens na Publicação MQTT.	57
5.11	<i>Log</i> de Subscrição em Todos os Tópicos via Mosquitto.	57
5.12	<i>Log</i> do <i>Script</i> de Armazenamento de Dados no Banco SQL.	58
5.13	Gráfico de Tensão por Horário.	58
5.14	Gráfico de Corrente por Horário.	59
5.15	Gráfico Curva de Carga Versus Previsão de Carga.	60
5.16	Análise Estatística da Previsão de Carga Consumidora.	60

Lista de Tabelas

1.1 Plano de Desenvolvimento.	7
---------------------------------------	---

Lista de Siglas e Abreviaturas

Abreviatura	Descrição
ADC	<i>Analog-to-Digital Converter</i>
ASK	<i>Amplitude-shift Keying</i>
CSS	<i>Chirp Spread Spectrum</i>
EUA	Estados Unidos da América
FSK	<i>Frequency-shift Keying</i>
IMA	Infraestrutura de Medição Avançada
IoT	<i>Internet of Things</i>
LPWAN	<i>Low Power Wide Area Network</i>
LSTM	<i>Long Short Term Memory</i>
MAPE	<i>Mean Absolute Percent Error</i>
MME	Ministério de Minas e Energia
MQTT	<i>Message Queuing Telemetry Transport</i>
M2M	<i>Machine to Machine</i>
PLC	<i>Powerline Communication</i>
PSK	<i>Phase-shift Keying</i>
REI	Rede Elétrica Inteligente
RMS	<i>Root Mean Square</i>
RNA	Redes Neurais Artificiais
RNR	Redes Neurais Recorrentes
ROM	<i>Read Only Memory</i>
SEP	Sistema Elétrico de Potência
SRAM	<i>Static Random Access Memory</i>

Agradecimentos

Instituto Superior de Engenharia do Porto (ISEP)

Universidade Presbiteriana Mackenzie (UPM).

Antonio Newton Licciardi Junior.

Ramiro Barbosa.

Família.

Amigos.

Capítulo 1

Introdução

Neste capítulo, visa-se apresentar ao leitor uma contextualização a respeito do tema *smart grid*, é apresentado uma breve história sobre o controlo do homem frente a energia elétrica, assim como a criação das redes elétricas e seus sistemas de medição. Posteriormente, é apresentado a justificativa e os objetivos da presente dissertação. Por fim, é relatado sobre o cronograma do projeto e a estrutura da dissertação.

1.1 Contexto

A história da humanidade pode ser marcada por algumas inovações, descobertas e invenções científicas que revolucionaram a rotina e o cotidiano de toda a sociedade, o domínio da energia elétrica de fato está presente nessa lista.

No início do século XX, o controlo da energia elétrica facilitou a iluminação de ambientes externos e internos das cidades, aumentou a segurança, facilitou o trabalho, estudo e a urbanização de grandes centros. No ramo da indústria, a energia elétrica, devido às suas características de transmissibilidade de grande quantidade de energia e também, sua fácil conversibilidade em movimento, luz e calor, foi um dos principais fatores responsáveis pela segunda revolução industrial.

A energia a vapor exigia que os geradores fossem instalados na própria fábrica, e mesmo assim, dentro de sua transmissão interna ainda ocorriam perdas significativas de energia. A energia elétrica possibilitou que os motores fossem acoplados junto aos instrumentos, diminuindo o uso de eixos e correias de transmissão, e consequentemente diminuindo a perda de energia. Além disso, facilitou a surgimento de pequenas e médias indústrias, pois a fonte de geração

de energia podia agora ser compartilhada entre muitas fábricas, sendo cobrado apenas a sua parcela da energia consumida [15].

Ao longo das décadas, o uso da energia elétrica também possibilitou a invenção de diversos dispositivos e equipamentos essenciais para o mundo moderno, como baterias, resistências, condensadores, indutâncias, dínamos, transformadores, motores elétricos, entre outros. Posteriormente, esses dispositivos foram utilizados para a criação de diversos bens de consumo: rádio, televisão, chuveiro elétrico, micro-ondas, computador, utensílios que aumentaram o conforto, a comunicação e a segurança de toda a sociedade.

Atualmente, o uso da energia elétrica já está completamente difundido pelas cidades, sendo indispensável para o mundo moderno. Sendo assim, a correta distribuição e a qualidade da energia elétrica são fatores determinantes para os avanços econômicos e sociais de uma região.

O cabeamento das grandes cidades teve início por volta de 1890 [16], inicialmente, a energia elétrica era gerada próxima aos centros de consumo, porém, devido ao grande aumento no consumo domésticos e industrial, tornou-se necessário o uso de energia de larga escala com grandes redes de distribuição [17]. Desde então, a infraestrutura da rede elétrica sofreu várias alterações, seja nos materiais utilizados, formas de cabeamento ou na própria topologia da rede, sempre visando a diminuição dos desperdícios, o aumento da segurança e a melhor adequação às necessidades energéticas de cada região.

Nos dias de hoje, o rápido avanço tecnológico, a facilidade de comunicação e a conscientização da população a respeito de questões ambientais, provocou alterações em diversos setores da sociedade. Pensando nesses novos paradigmas, as redes elétricas vêm-se atualizando a conceitos e tecnologias mais modernas. Nesse contexto, surgiu a ideia de *smart grid* ou redes inteligentes, que visa melhorar a eficiência energética, aumentar a automação e facilitar a troca de informação dentro do sistema elétrico.

Segundo Sergio Date [18], a implantação de redes inteligentes implica em uma melhoria no sistema de energia elétrica, acarretando em uma maior eficiência energética. Para isso, o contexto de *smart grid* engloba as ideias de medição eletrônica, telecomunicação, automação, ferramentas de sensoriamento e capacidade computacional, além de estruturas de fornecimento de energia elétrica a veículos elétricos.

O conceito de *smart grid* é muito abrangente, envolvendo diversas áreas do sistema elétrico, partindo na fase de geração até chegar ao consumidor final. De uma forma mais geral, o conceito de *smart grid* pode ser definido como uma rede de transmissão de energia e informação de forma bidirecional entre os consumidores e as concessionárias, visando a formação de uma rede inteligente que

possibilite uma maior eficiência e um melhor controlo da energia transmitida [19].

Dessa forma, para viabilizar esta rede integrada que permita a comunicação bidirecional entre as concessionárias e os consumidores, foi desenvolvido um equipamento de medição inteligente, que fornece primordialmente as leituras dos parâmetros da rede elétrica, e a transmissão desses dados de maneira confiável ao longo da rede.

A ideia da utilização de medidores no sistema elétrico surgiu devido a necessidade de contabilizar a energia consumida por cada cliente. No ano de 1881 Thomas Alva Edison que introduziu o primeiro sistema de distribuição elétrica patenteou seu medidor de energia, que utilizava os efeitos eletroquímicos causados pela passagem de corrente elétrica.

Na época, o medidor de Edison continha uma célula eletrolítica que dentro era colocado uma placa de cobre precisamente pesada no início do período de cobrança, a corrente elétrica passada através da célula eletrolítica realizava o processo de eletrólise na placa, causando o depósito de cobre no fundo do equipamento de medição. No final do período de cobrança, a diferença de peso na placa de cobre determinava a quantidade de energia utilizada pelo consumidor [20].

Durante muitos anos, a técnica de medição mais utilizada foi a da indução eletromagnética, que é implementada em conjunto de um medidor eletromecânico. Como pode ser visto em maiores detalhes em Agustín Mínguez [21]. Essa técnica utiliza o princípio da indução eletromagnética, a qual afirma que, um condutor percorrido por uma corrente I na presença de um campo magnético B , fica submetido a uma força F . Esse princípio é utilizado para construção de um medidor eletromecânico que é composto por um disco que permite rotação, bobinas que geram o campo magnético, um registrador que realiza as leituras do número de rotações do disco e um equipamento frenador, normalmente um íman, utilizado para calibração do medidor.

Como pode ser observado, durante as últimas décadas várias técnicas e equipamentos de medição foram desenvolvidos com o intuito de diminuir o custo, melhorar a precisão, facilitar as leituras e diminuir as manutenções. Porém, até recentemente os medidores mantinham a necessidade da realização de leituras locais para a cobrança das contas de energia.

O advento da eletrônica digital e o avanço das telecomunicações, permitiram a implementação de equipamentos mais sofisticados, os chamados *smart meters*, que permitem a leitura dos parâmetros em tempo real, retirando a necessidade de leituras locais para cobranças e permitindo inúmeras outras automações para o sistema elétrico.

De acordo com Sergio Date [18], um *smart meter* pode ser definido como um dispositivo de medição baseado em diferentes princípios de funcionamento, como elétrico, eletrônico e mecânico, porém, cada vez mais implementado com sistemas eletrônicos e digitais que tornam uma ferramenta interativa.

Segundo Depuru [22], um *smart meter* é um medidor de energia que mede o consumo de energia elétrica do consumidor, além disso, provê outras informações adicionais para as concessionárias quando comparado aos medidores tradicionais. Sendo assim, sua integração na rede elétrica envolve uma variedade de técnicas de *software*, que dependem das necessidades da demanda do consumidor. Além disso, a implantação de um *smart meter* necessita de uma seleção de rede de comunicação adequada, assim satisfazendo os padrões de segurança da rede inteligente.

1.2 Justificativa

Como comentado na introdução, a implementação de uma rede *smart grid* com tráfego de dados em tempo real oferece diversos benefícios tanto para as concessionárias de energia quanto para os consumidores. De acordo com Cabello [23], a implementação de um rede de *smart grid* gera benefícios como:

- Serviços mais eficientes a um custo menor;
- Melhor detecção e, consequentemente, resposta mais rápida a eventuais problemas;
- Redução do consumo por meio de uma melhor gestão do uso de energia;
- Consumidor com maior controlo sobre seus gastos;
- Melhor monitoramento da rede possibilita uma queda expressiva nas perdas sofridas por fraude ou roubo de energia;
- Maior espaço para fontes alternativas de geração distribuída, devido a bidirecionalidade da rede.

Vale-se ressaltar que a redução do consumo por meio de uma melhor gestão do uso de energia pode ser alcançada utilizando modelos de inteligência artificial, com estes prever a demanda de um consumidor ou de uma região. Com esses modelos, torna-se possível dimensionar de forma mais precisa as necessidades energéticas e os horários de pico de cada região ou consumidor individual.

Dito isso, torna-se nítido que a implementação de uma rede elétrica inteligente (REI) representa um avanço para o setor elétrico, consequentemente, para

a sociedade como um todo. Sendo assim, para facilitar a difusão dessa tecnologia, são necessários estudos e pesquisas relacionadas ao tema, para assim: examinar, guiar e padronizar essa nova tecnologia em forte ascensão.

Devido ao fato de a tecnologia de *smart grid* ser ainda incipiente, existem diversas características técnicas a serem discutidas para sua melhor implementação em larga escala. Uma dessas questões debatidas é sobre a definição da tecnologia de telecomunicação mais adequada para sua implementação.

Atualmente, uma das tecnologias de telecomunicações mais promissora é a chamada *Powerline Communication* (PLC). A tecnologia PLC possui a vantagem de utilizar a própria infraestrutura já instalada do sistema elétrico de potência para trafegar os dados necessários para o funcionamento do *smart grid* [24].

De acordo com Augusto Matheus [4], alguns dos pontos mais fortes desta tecnologia são destacados por:

- Acesso ilimitado a toda abrangência do sistema de transmissão de energia;
- Utilização da mesma estrutura física dos *grids* de energia eliminando a necessidade de novo cabeamento;
- Habilidade de oferecer altas taxas de aquisição de dados para estabelecimento de redes de comunicação;
- Divisão de faixas de frequência para transmissão específica de dados;
- Combinação com outros tipos de tecnologia.

Apesar da tecnologia PLC oferecer diversas facilidades, existem alguns fatores que vêm dificultando sua implementação nas REI. Devido à utilização do SEP para tráfego de dados, o PLC enfrenta um ambiente instável, sujeito a interferências eletromagnéticas externas, assim como, ruídos e variações internas à rede.

Segundo Márco Feliciano [25], existem algumas desvantagens no uso do PLC que merecem ser analisadas, como:

- Atenuação de acordo com a frequência: divisores de tensão, acoplamento entre fases;
- Atraso na comunicação;
- Ruído impulsivo: provenientes motores, *dimmers*, interruptores;
- Falta de segurança;
- Problemas de compatibilidade eletromagnética, devido ao interfaceamento de circuitos eletrônicos com o SEP;

- Imprevisibilidade de parâmetros do canal utilizado, como: impedância, atenuação e níveis de ruídos, que oscilam com o tempo e a alteração de carga na rede.

Sendo assim, torna-se interessante estudos a respeito de outras formas de comunicações alternativas para serem utilizadas nas REI. Uma dessas tecnologias de comunicação que está tendo grande destaque atualmente são as chamadas *Low Power Wide Area Network* ou apenas LPWAN (em tradução livre, Redes de longa distância e baixo consumo).

Segundo Câmera da Silva [26], as redes LPWAN permitem conectar dispositivos por longas distâncias (e.g., de três a cinquenta quilômetros) com baixo custo (e.g., um *end node* pode chegar a cinco dólares), com pequena largura de banda e, consequentemente, baixo consumo relativo de bateria.

Dessa forma, devido aos sistemas *smart grid* não necessitarem de um alto tráfego de dados e de as redes LPWAN proporcionarem um baixo custo e longo alcance, fazem destas um candidato viável como tecnologia de telecomunicação dentro de um sistema de REI.

1.3 Objetivos

Esse trabalho tem como objetivo principal de realizar um protótipo de baixo custo de um sistema de *smart grid* completo, utilizando como forma de comunicação uma rede LPWAN. Assim como, visa também adicionar algumas funcionalidades não sempre exploradas dentro do medidor inteligente, sendo estas: cálculo da frequência da rede e cálculo do fator de potência da instalação.

Além disso, o presente trabalho pretende também realizar testes no protótipo final, com a finalidade de demonstrar a funcionalidade e confiabilidade de um sistema de REI que utilize uma rede LPWAN para seu tráfego de dados.

Por fim, a presente dissertação visa pesquisar e definir um modelo de inteligência artificial adequado para realizar a previsão de demanda do consumidor.

Para realização destes objetivos, faz-se necessário o detalhamento de algumas etapas do projeto, dentre elas:

- Construção de protótipo de medidor inteligente de baixo custo com a tecnologia LoRaWAN para comunicação;
- Adição da funcionalidade de fator de potência e frequência da rede no protótipo do medidor inteligente;
- Construção de protótipo de *gateway* LoRa para recepção dos dados enviados pelo medidor inteligente;

- Configuração de servidor MQTT para receber os dados enviados pelo *gateway LoRa*;
- Codificação de um *script* ou rotina que armazene os dados do servidor MQTT em um banco de dados;
- Construção de *dashboard* para visualização e análise dos dados medidos pelo *smart meter*;
- Estudo e implementação de modelo de inteligência artificial para previsão de demanda de carga do consumidor.

1.4 Calendarização

Para elaboração do presente trabalho, divide-se em etapas os objetivos realizados. Inicialmente, fez-se necessário o aprofundamento na bibliografia disponível sobre o tema, para assim, definir-se a tecnologia a ser utilizada na rede LPWAN, assim como, os componentes e dispositivos necessários para realização do protótipo da REI.

Posteriormente, é realizado a compra dos componentes necessários e inicia-se a fase de prototipagem do medidor inteligente. Com o *smart meter* funcional inicia-se a elaboração da dissertação e a pesquisa e configuração dos *softwares* necessários para o funcionamento da comunicação entre *smart meter* e servidor.

Seguido a finalização do sistema de *smart grid* completo, ou seja, a correta leitura e tráfego de dados entre *smart meter* e servidor, inicia-se a fase de pesquisa e definição do modelo de inteligência artificial mais adequado para a previsão de carga de um consumidor ou região.

Por fim, são realizados os testes de funcionalidade e confiabilidade no protótipo final, assim como, a elaboração da versão final da presente dissertação.

A visualização gráfica do plano de calendarização pode ser observada na Tabela 1.1.

Tabela 1.1: Plano de Desenvolvimento.

	Abr.	Maio	Jun.	Jul.	Ago.	Set.	Out.
Aprofundamento no tema	X						
Orçamento do <i>hardware</i>		X					
Prototipagem de <i>hardware</i>		X	X	X			
Desenvolvimento de <i>software</i>				X	X		
Testes no protótipo					X	X	
Relatório			X	X	X	X	X

1.5 Estrutura

Inicialmente, esta dissertação apresenta a contextualização sobre o tema, para isso, oferece uma breve introdução de como o homem vem utilizando a energia durante a história, assim como, um contexto sobre o surgimento das redes elétricas atuais e seus medidores de energia elétrica.

Seguido a contextualização, a justificativa apresenta as vantagens sobre estudos a respeito do tema e também das necessidades de formas de comunicações alternativas para implementação de REI em larga escala.

Posteriormente, são apresentados os objetivos principais da dissertação, assim como, as etapas necessárias para alcançar tais objetivos. Após isso, é descrito brevemente o plano de desenvolvimento da dissertação, tal como, o presente tópico, a estrutura da mesma.

Seguido da fase introdutória, inicia-se o capítulo de “Estado da Arte”, o qual visa trazer ao leitor o atual nível de desenvolvimento das técnicas e tecnologias que dizem respeito ao tema, assim como, a revisão bibliográfica utilizada para presente pesquisa.

No capítulo “Prototipagem de *Hardware*” é detalhado a construção dos protótipos: *smart meter* e *gateway*. Além disso, neste capítulo é examinado a qualidade e precisão dos componentes escolhidos para utilização no protótipo.

Seguido da “Prototipagem de *Hardware*” inicia-se o capítulo de “Arquitetura de *Software*”. Este capítulo visa detalhar os *softwares* e os códigos necessários para a elaboração do sistema *smart grid* de baixo custo. Além de que, este capítulo visa também detalhar o modelo de inteligência artificial escolhido como mais adequado para análise de previsão de demanda dos consumidores da rede elétrica.

Após o capítulo “Arquitetura de *Software*”, inicia-se a fase de testes no protótipo final, tendo a finalidade de apresentar e discutir os resultados obtidos, tal como, fornecer ao leitor algumas vantagens e desvantagens encontradas durante os processos de prototipagem e modelagem do sistema *smart grid*.

Por fim, no capítulo “Conclusões”, é ponderado algumas constatações a respeito dos resultados obtidos, tal como, as principais dificuldades encontradas e algumas ideias para trabalhos futuros.

Capítulo 2

Estado da arte

Este capítulo visa contextualizar o leitor a respeito do atual nível de desenvolvimento das tecnologias: *smart grid*, *smart meter*, redes LPWAN, protocolo de comunicação MQTT e modelos de inteligência artificial para análise de consumo no setor elétrico. Além disso, neste capítulo, pretende-se apresentar ao leitor as referências utilizadas para a construção do protótipo apresentado na presente dissertação.

2.1 *Smart Grid*

2.1.1 Cenário Atual dos Sistemas de Rede Elétrica

Um sistema de rede elétrica tem como objetivo principal o fornecimento de energia aos consumidores finais. Para isso, um SEP se encarrega dos processos de geração, transmissão e distribuição da energia elétrica. A topologia de um SEP pode ser vista na Figura 2.1.

Atualmente, com a revolução digital, a facilidade de tráfego de dados e informações possibilitou a implantação de sistemas de monitoramento e segurança em diversos serviços. Dito isso, o SEP exige mudanças para sua correta adequação ao mundo moderno.

De acordo com Joberto Martins [3], as experiências têm mostrado que a obsolescência dos sistemas da rede elétrica do século 20 não se adapta mais às necessidades do século 21, que incluem não apenas um aumento na demanda por recursos, mas também um aumento na qualidade e na variedade dos serviços providos.

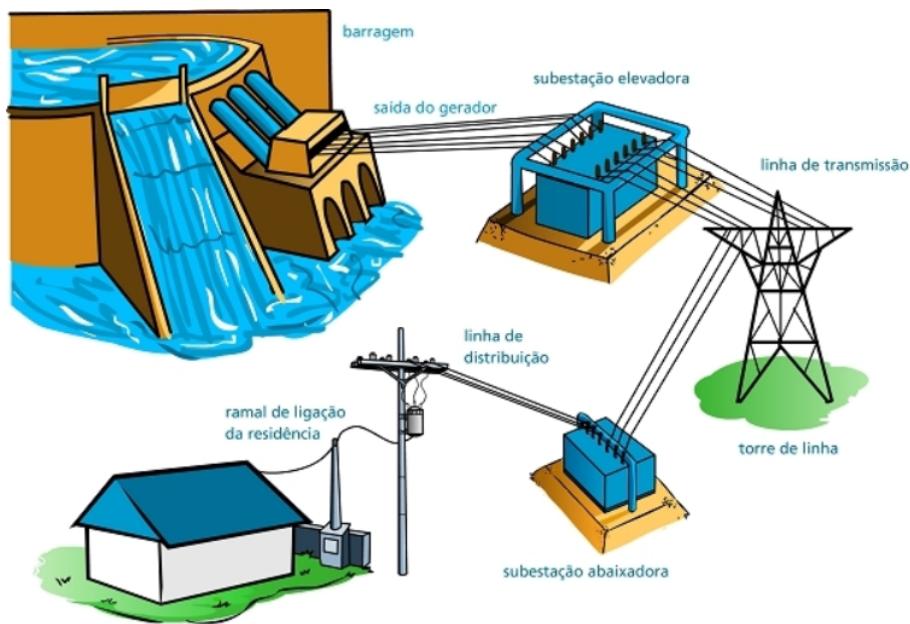


Figura 2.1: Topologia de um Sistema Elétrico de Potência [1].

Algumas funções e requisitos que os SEP atualmente apresentam dificuldade em fornecer são:

- Integração de novas demandas (como por exemplo, veículos elétricos, e gerações distribuídas);
- Nível de automação adequado no controlo dos dispositivos da rede;
- Detalhamento preciso dos dados sobre o consumo de energia;
- Segurança contra furtos (de equipamento e de energia);
- Informações detalhadas a respeito do consumo de energia para os clientes;
- Qualidade adequada na energia entregue ao consumidor (devido a falhas nos sistemas de transmissão e de distribuição);
- Controlo automático do consumo de energia.

2.1.2 *Smart Grid ou Redes Inteligentes*

Neste contexto, com o propósito de eliminar alguns dos problemas e obsolescências das redes elétricas surgiram soluções como o *smart grid*. O *smart grid*

ou REI se caracteriza pela integração de uma cadeia de informação dentro do sistema elétrico de potência. Essa cadeia de informação possibilita a implementação de diversos benefícios tanto aos geradores e distribuidores como para os consumidores da energia.

Se tratando da topologia de uma rede elétrica inteligente, esta implementa uma infraestrutura externa à rede elétrica, que deve ser capaz de lidar com o fluxo de informação bidirecional dentro da rede. A estrutura e topologia de uma REI pode ser vista na Figura 2.2.

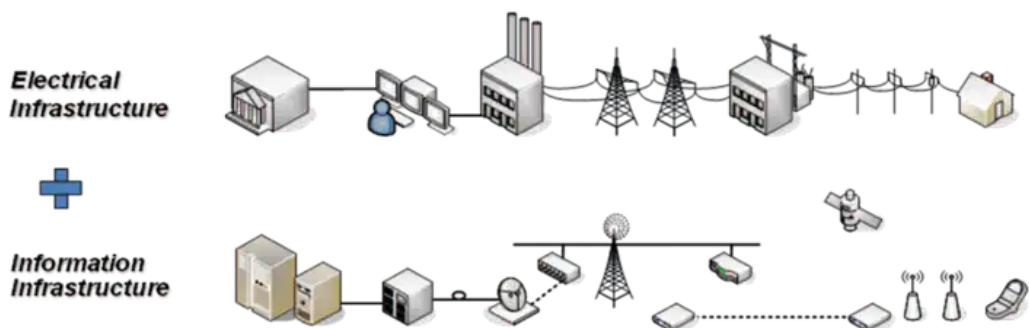


Figura 2.2: Topologia de um Sistema *Smart Grid* [2].

Dentre os benefícios oferecidos pela implantação de uma REI, pode-se citar a possibilidade de monitoramento do tráfego de energia, permitindo assim a concessionária identificar falhas, fraudes ou furtos durante as fases de distribuição e transmissão.

Segundo Paulo Barnabé [27], o aprimoramento dos medidores inteligentes na rede de distribuição não exclui por completo o número de fraudes e furtos de energia, contudo, esta nova tecnologia traz consigo a característica inibidora de atividades ilícitas contra o sistema de medição uma vez que a adulteração no equipamento ou na linha seria de rápida e fácil detecção tanto para a unidade consumidora violada quanto para os agentes da distribuidora.

De acordo com Danielle Marques [28], uma vez que existe a probabilidade de ocorrência de falhas em uma rede elétrica e riscos associados a esses eventos, ter informações precisas torna-se fundamental para o processo de tomada de decisão.

Se tratando do consumidor, este melhor monitoramento e resposta rápida às falhas prometem uma diminuição nas perdas de energias, consequentemente, uma diminuição no preço da energia. Além disso, um sistema de *smart grid* permite ao consumidor um maior acesso a informações de consumo e preço, permitindo assim o consumidor se adequar aos melhores horários de consumo.

Conforme dito por Ricardo Rivera [29], do ponto de vista dos consumidores, podem-se obter: informação sobre o consumo de energia por horário – tarifa branca; apresentação de dados do último período de faturamento (memória de massa); e indicativos da qualidade da energia ofertada pelas concessionárias, permitindo que a agência reguladora possa, por exemplo, reduzir o valor cobrado pela energia caso os indicadores fiquem fora do padrão de qualidade estabelecido.

Vale-se destacar, que com o aumento no monitoramento da rede, uma REI visa permitir um fluxo bidirecional não apenas da comunicação como também da energia. Essa bidirecionalidade permite a implementação de geradoras distribuídas ao longo de toda a cadeia. A geração de energia distribuída encurta o caminho entre o gerador e o consumidor, consequentemente, diminui as perdas de energia durante a transmissão e distribuição.

Além disso, a produção de energia distribuída em larga escala, abre possibilidade para um novo cenário de mercado livre de energia, onde permite o consumidor escolher a geradora de preferência para compra da energia. Dessa forma, o aumento da concorrência pode acarretar em uma diminuição nos preços das tarifas e aumento na qualidade da energia distribuída.

2.1.3 *Microgrids*

No escopo da geração distribuída, um dos principais paradigmas sendo discutidos são os *microgrids*. Os sistemas de *microgrids* são caracterizados pela instalação de uma ou mais fontes geradoras em uma área delimitada, e o isolamento facultativo desta área da rede principal de distribuição elétrica. Assim sendo, é possível conectar pequenas usinas de fontes renováveis a um sistema local.

Como nos assegura Fang Yang [30], *microgrids* é um sistema de energia integrado, que inclui um grupo de recursos energéticos distribuídos e interconectados localizados dentro de uma área com seus limites elétricos claramente definidos. Dessa maneira, o sistema funciona como um SEP de pequena escala, podendo operar de maneira conectado à principal rede ou separado da rede principal.

O isolamento facultativo permite a região delimitada ficar isoladas aos problemas advindos da rede nos momentos que sua fonte geradora estiver a suprir completamente a demanda energética. Por outro lado, o isolamento facultativo também permite a área delimitada se conectar à rede em momentos de falta de oferta energética.

Dessa forma, a interconexão de diversas *microgrids*, pode-se criar uma *smart grid* mais segura e independente, na qual a transmissão da informação siga um

caminho diferente do fluxo de energia, como pode ser visto na Figura 2.3.

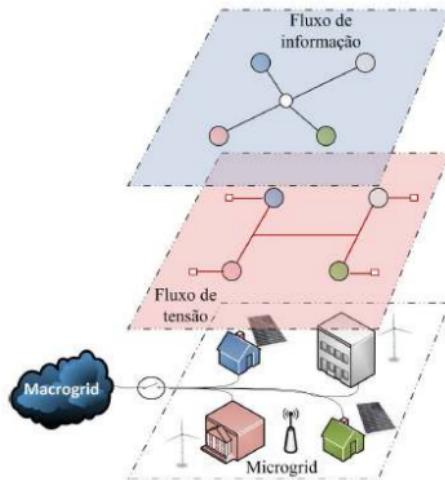


Figura 2.3: Exemplo de *Microgrids* [3].

2.1.4 Experiências de Implementação

Durante os últimos anos, iniciaram projetos de redes elétricas inteligentes em todo o mundo. Vários países e empresas multinacionais estão realizando investimentos em cidades ou regiões com o intuito de pesquisar e averiguar a viabilidade prática deste tipo de sistema [23].

De acordo com João Leite [31], as redes inteligentes de energia já são uma realidade em nível internacional, nos EUA em 2008 foi aprovada a legislação sobre redes inteligentes e em 2009, o montante de 4 bilhões de dólares foi inserido em um pacote de incentivos destinado ao desenvolvimento dessas redes, por meio dos programas: *Smart Grid Investment Grant* (SGIG) e *Smart Grid Demonstration Program* (SGDP).

Em março de 2008, a cidade de Boulder no Colorado, foi escolhida como a sede do projeto desenvolvido pela estatal Xcel Energy para se tornar a primeira cidade do mundo com a tecnologia de *smart grid* realmente implantadas. A escolha da cidade de Boulder foi devido ao seu tamanho, sua rede de infraestrutura e sua proximidade com instituições como a Universidade do Colorado [32].

Conforme Cássia Pereira [33], centenas de projetos estão em desenvolvimento nos Estados Unidos, dos quais a maioria focados em Infraestrutura de Medição Avançada (IMA), que visa principalmente a implantação de medido-

res avançados, incluindo monitoramento contra furtos de energia, serviços de operação e atendimento ao cliente.

Segundo Cabello [23], na Europa, praticamente todos os países já têm alguma iniciativa relacionada a redes inteligentes. Tanto a Itália quanto a Malta têm projetos de escala nacional. Outros países, como Alemanha, Portugal e Reino Unido, já estão testando a tecnologia em escala menor para avaliar seu potencial em termos de economia de recursos.

2.1.5 Desafios

A implementação de um sistema de *smart grid* exige um grande empenho e investimento no setor elétrico, muito disso devido à grande infraestrutura e a vasta extensão territorial que o sistema elétrico normalmente exige. Como informa Joberto Martins [3], não houve mudanças revolucionárias na estrutura da rede de energia elétrica nos últimos 100 anos. Essa constatação demonstra a dificuldade para implementação de mudanças neste setor.

Como bem destaca Maria Ferreira [32], não se pode esquecer de que é fundamental a vontade política para as mudanças em direção a implementação destas novas tecnologias (*smart grids*) aconteçam de forma rápida e eficiente, e que quanto mais complexa a mudança proposta mais difícil é que haja um consenso para a sua aplicação, tornando o processo ainda mais demorado e custoso.

No contexto de *microgrids*, de acordo com Yeliz Yoldas [34], devido a capacidade de transição entre conexão e desconexão à rede principal, o sistema pode causar uma grande incompatibilidade, levando a graves problemas no controlo de frequência e tensão entre os geradores e as cargas.

2.1.6 Perspectiva para o Futuro

O Brasil possui atualmente várias iniciativas para implementação de redes inteligentes, essas iniciativas caracterizadas principalmente pela substituição de medidores eletromecânicos por medidores digitais, com foco em reduzir as perdas por furtos de energia.

De acordo com Maria Ferreira [32], Portugal tem tido iniciativas interessantes em direção da implantação das REI, além disso por ser um país europeu terá que diminuir as suas emissões de CO₂ em 20 porcento em relação aos níveis de 1990, conforme foi acordado pela União Europeia, e neste contexto as redes *smart grid* se colocam como uma solução tanto na possibilidade de utilizar energias limpas como na melhoria da eficiência energética.

Segundo Miguel da Silva [35], a cidade de Évora foi selecionada para a implementação do InovCity, com base em critérios de dimensão, características eléctricas da rede e visibilidade da cidade. O InovCity representa o conceito das

redes inteligentes de eletricidade, enquanto piloto da primeira *smart grid* Ibérica, tendo como objetivos: contribuir efetivamente na redução de emissões de CO₂ e aumentar a contribuição das energias renováveis. Dessa forma, aumentando a eficiência energética do setor elétrico.

2.2 Smart Meter

2.2.1 Smart Meter ou Medidor Inteligente

Como destacado anteriormente, um dos dispositivos fundamentais para uma REI é o medidor inteligente. Usualmente, essa nova tecnologia para medição elétrica, além de oferecer medições mais precisas, também deve possibilitar um canal de comunicação entre o operador da rede e as cargas da unidade consumidora.

Esta nova tecnologia tem ganhado amplo espaço dentro do ramo de medições elétricas, sendo hoje utilizado não apenas por unidades consumidoras como também em subestações, pontos de intercâmbio de energia e sistemas de transmissões de média e alta tensão. Sendo este dispositivo comum para medição de grandes blocos de energia [18]. Na Figura 2.4 pode ser observado um exemplo de *smart meter*.



Figura 2.4: Exemplo de *Smart Meter* [4].

2.2.2 Características e Funcionalidades

Assim como comentado anteriormente, não existe uma definição exata para o conceito de *smart meter*, sendo suas características fundamentais: capacidade

de medição elétrica e comunicação entre concessionária e consumidor. Dito isso, com a finalidade de oferecer benefícios a operação da rede elétrica, uma gama de outras funcionalidades podem ser elaboradas junto a um dispositivo de medição inteligente, sendo algumas funcionalidades mais simples e outras mais complexas de serem implementadas.

Atualmente, algumas das principais funcionalidades implementadas pelos medidores inteligentes são:

- Apresentação da potência instantânea e custo do kWh em um *display*;
- Comunicação bidirecional;
- Operação de ligamento e desligamento da unidade consumidora;
- Medição da frequência da rede elétrica;
- Análise da qualidade da energia na rede elétrica;
- Medição do fator de potência da unidade consumidora;
- Medição das interrupções de energia (quantidade e tempo).

Embora existam tipos e funcionalidades diferentes de *smart meters*, a finalidade de sua implantação é a melhora na qualidade do fornecimento de energia. Com isso, gerar economias tanto para a parte do consumidor como para as concessionárias, dessa forma, contribuindo na eficiência energética [18].

2.3 LPWAN, *Low Power Wide Area Network*

2.3.1 Conceitos de modulação

Modulação é o processo pelo qual uma onda periódica, também chamada onda portadora, é combinada junto a uma outra onda que contém o sinal com a informação que deseja ser transmitida. Todo o sinal contém três características que podem ser moduladas: frequência, amplitude e fase. Frequência da onda define a quantidade de ciclos completos em um determinado tempo, amplitude define a “força” ou energia do formato de onda, fase define o estado do formato de onda em um respectivo tempo em um determinado ciclo [5].

A forma de onda final, formada da junção da onda portadora e do sinal de dados recebe o nome de símbolo. A taxa de bits é definida como o número de bits enviados pelo sistema por unidade de tempo. A taxa de símbolos é dada

pela divisão da taxa de bits pelo número de bits sendo enviado em cada símbolo [5].

Tratando-se da modulação analógica, as suas principais formas de modulação são:

- Modulação de frequência: A modulação de frequência tem vasta utilidade em rádios FM, radares, telemetrias e sistemas de armazenamentos de fitas magnéticas. Um exemplo de modulação de frequência pode ser observado na Figura 2.5.

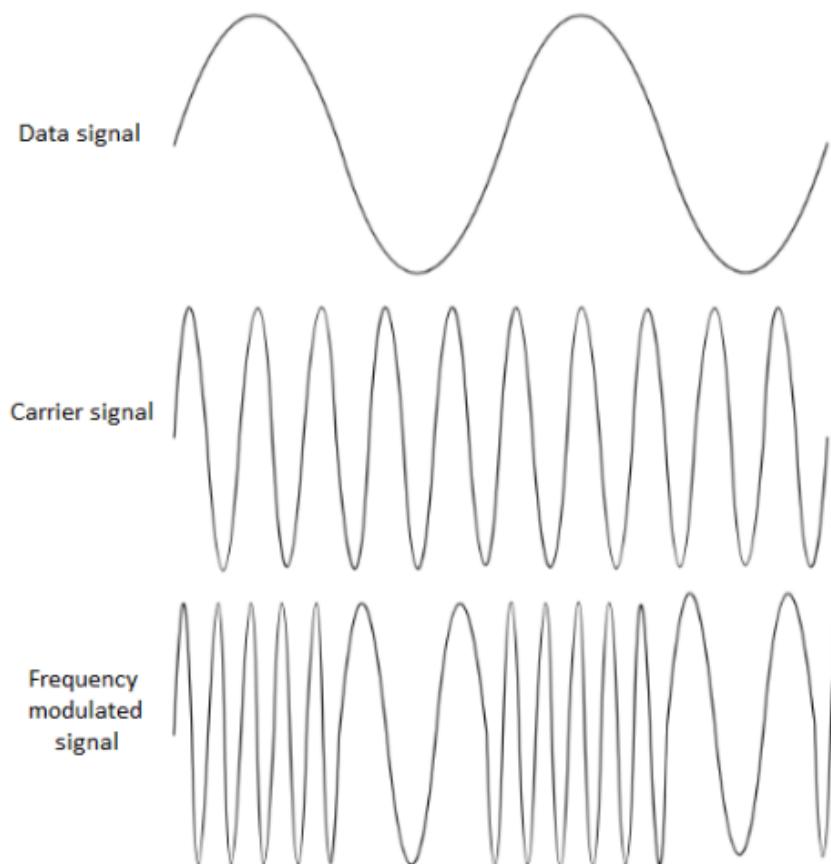


Figura 2.5: Exemplo de Modulação de Frequênciа [5].

- Modulação de amplitude: A modulação de amplitude é muito utilizada para rádios AM. Um exemplo de modulação de amplitude pode ser observado na Figura 2.6.

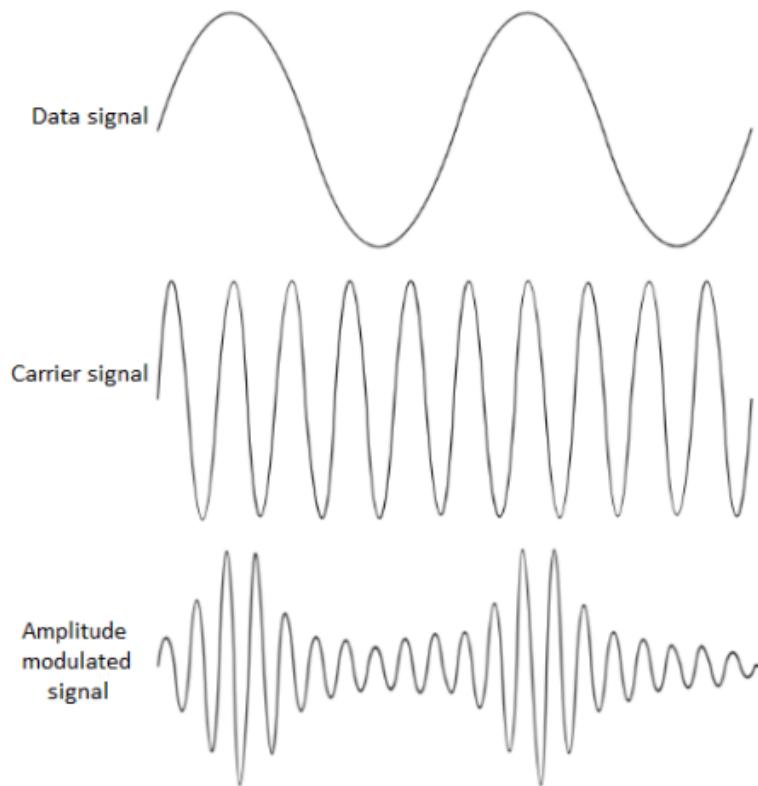


Figura 2.6: Exemplo de Modulação de Amplitude [5].

No que diz respeito a modulação digital, existem três principais esquemas de modulação utilizados para representar os valores zero e um:

- *Phase-shift Keying (PSK)*: A modulação por deslocamento de fase ou apenas PSK é um processo de modulação digital pelo qual o dado é transmitido alterando a fase da onda portadora conforme um sinal de referência com frequência constante. A modulação é realizada alterando as entradas de seno e cosseno em um tempo específico;
- *Frequency-shift Keying (FSK)*: A modulação por deslocamento de frequência ou apenas FSK é uma forma de modulação de frequência em que a informação digital é transmitida através de mudanças discretas na frequência da onda portadora;
- *Amplitude-shift Keying (ASK)*: A modulação por chaveamento de amplitude ou apenas ASK é uma forma de modulação de amplitude que representa o sinal digital conforme a variação de amplitude da onda portadora.

Na Figura 2.7 pode ser observado um exemplo de modulação de PSK, FSK e ASK para a sequência binária 0010100.

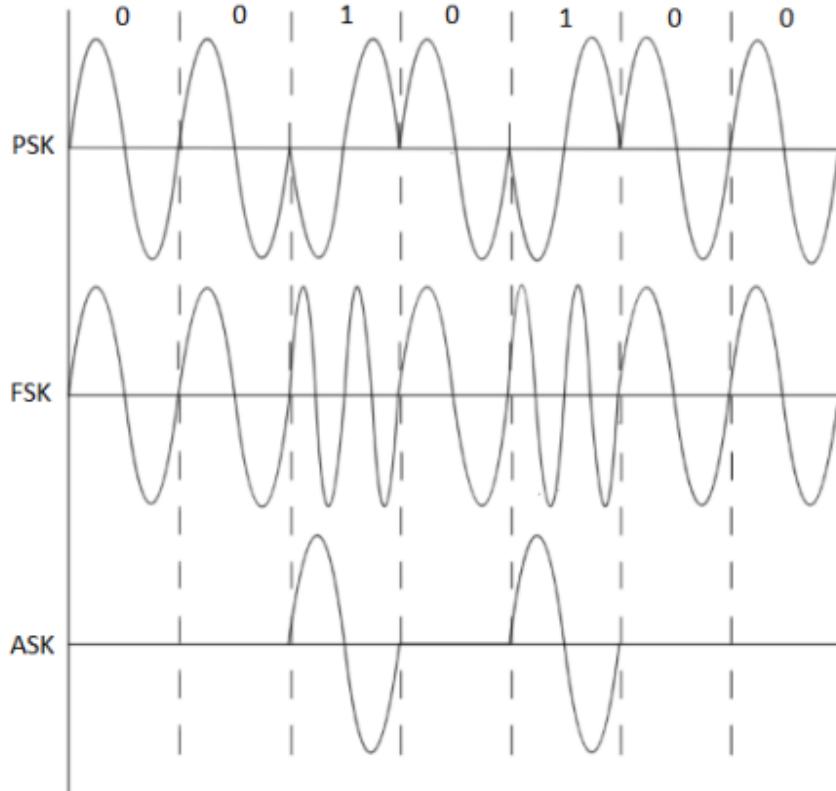


Figura 2.7: Exemplo de Modulação PSK, FSK e ASK [5].

2.3.2 Redes LPWAN

As chamadas redes LPWAN (*Low Power Wide Area Network*), ou em tradução livre, redes de longa distância e baixo consumo, são redes que se caracterizam pela capacidade de ter grandes áreas de cobertura, baixa taxa de transferência de dados e um baixo consumo de energia do equipamento.

As redes LPWAN podem chegar a uma distância de 5 km em áreas urbanas e 15 km em áreas rurais. Isso é possível devido ao novo projeto da camada física, o qual é desenhado para ter uma alta sensibilidade do receptor [36].

De acordo com Dhaval Patel [5], as redes LPWAN alcançam sua longa distância de comunicação com técnicas de modulação que focam em formas de ondas com alta energia por bit ao invés de formas de onda com alta taxa de bits. Sendo assim, a maioria das tecnologias LPWAN diminui suas taxas de modula-

ção com o objetivo de colocar mais energia em cada bit ou símbolo transmitido. Dessa forma, o receptor é capaz de descodificar até mesmo os sinais mais fracos sem erros.

Segundo Usman Raza [37], existe duas classes de técnicas de modulação que são adotadas pelas diferentes tecnologias que utilizam LPWAN:

- *Narrowband*: A tecnologia de modulação *Narrowband* prove um alto *link budget* (contabilidade dos ganhos e perdas de energia que um sinal de comunicação experimenta em um sistema de telecomunicações) devido a codificação do seu sinal em pequenas larguras de banda (usualmente menos de 25 kHz). Atribuindo em cada onda portadora uma banda estreita, esse tipo de técnica de modulação compartilha o espectro total de uma forma muito eficiente entre os múltiplos *links*. O nível de ruído experienciado em cada banda estreita também é mínimo;
- *Spread spectrum techniques*: A tecnologia de modulação *Spread spectrum techniques* espalha um sinal de banda estreita em uma banda de frequência maior, porém, com a mesma potência do sinal. Esta forma de transmissão é similar a um sinal de ruído, que a torna difícil de ser detectada, mais resiliente a interferências e mais robusta a ataques de congestionamento. Entretanto, um maior poder de processamento é requisitado ao receptor para conseguir descodificar a mensagem. Espalhar a o sinal da banda estreita em uma maior banda de frequência naturalmente implica em um uso menos eficiente do espectro de frequência. Entretanto, esse problema é tipicamente solucionado utilizando sequências ortogonais. Dessa forma, contanto que os diferentes dispositivos utilizem diferentes canais ou sequências ortogonais a descodificação pode ser realizada de forma concorrente.

2.3.3 LoRa e LoRaWAN

LoRa é uma tecnologia de camada física que modula o sinal em um SUB-GHZ ISM band usando uma técnica de *spread spectrum* proprietária [38], desenvolvida e comercializada pela Semtech Corporation [39]. Uma comunicação bidirecional é provida pela técnica de *chirp spread spectrum* (CSS), que espalha a banda estreita de um sinal de entrada em um maior canal de largura de banda. O resultado é um sinal com propriedades parecidas a de um ruído. Dessa forma, tornando o sinal difícil de ser detectado. Além disso, o ganho de processamento possibilita uma maior resiliência a interferências e ruídos [37].

O protocolo LoRaWAN define a arquitetura do sistema, parâmetros de comunicação, segurança, qualidade do serviço e ajuste de potência utilizados pela tecnologia LoRa. Este protocolo permite a operação nas bandas 433, 868, 915 MHz dependendo da região. A largura do canal é definida como 125 kHz. A

modulação geralmente utilizada é a CSS, sendo também compatível com FSK e GFSK. A taxa e dados suportada é entre 0.3 kbps e 50 kbps [40].

A tecnologia LoRa utiliza a comunicação *half-duplex*, ou seja, é estabelecido uma conexão bidirecional entre as partes, porém, é permitido apenas o fluxo de dados em uma direção por vez. Em outras palavras, não existe transmissão simultânea, em um momento apenas o rádio A transmite, em outro momento apenas o rádio B transmite. Devido a essa característica *half-duplex* e também um maior controlo sobre o consumo de energia, o protocolo LoRaWAN implementa 3 classes de operações para os rádios:

- Classe A: Dispositivos classe A têm apenas 2 curtas janelas de recepção após a transmissão de um pacote. Após a transmissão do pacote, os dispositivos classe A entram em modo repouso com a intenção de conservar energia, como pode ser observado na Figura 2.8.

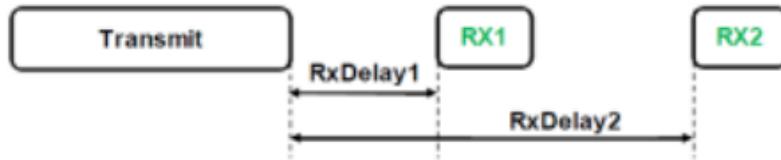


Figura 2.8: Janela de Recepção de Dispositivos Classe A [6].

- Classe B: Similar a janela de recepção dos dispositivos classe A, os dispositivos classe B possuem uma janela de recepção adicional, com intervalos agendados. Essa janela de recepção é sincronizada pelos *beacons* enviados pelo *gateway*, como pode ser observado na Figura 2.9.

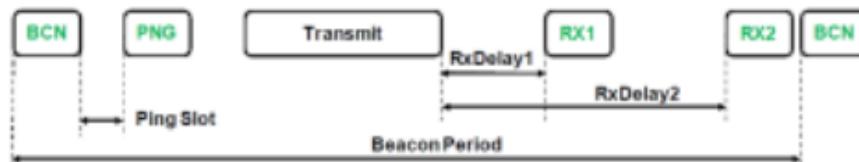


Figura 2.9: Janela de Recepção de Dispositivos Classe B [6].

- Classe C: Dispositivos classe C normalmente não são alimentados por baterias, o que permite manter o rádio de comunicação continuamente em modo de recepção, como pode ser observado na Figura 2.10.

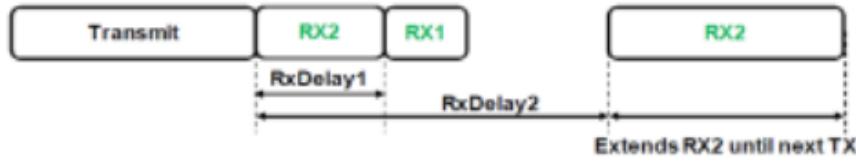


Figura 2.10: Janela de Recepção de Dispositivos Classe C [6].

Devido a essa limitação da característica *half-duplex*, apesar de incomum, algumas aplicações com rádio LoRa que necessitam de uma comunicação *full-duplex*, utilizam dois pares de rádios paralelos, permitindo um canal apenas para transmissão e outro canal apenas para recepção dos dados. Dessa forma, se estabelece uma comunicação bidirecional simultânea entre os dispositivos.

2.4 MQTT - *Message Queuing Telemetry Transport*

2.4.1 Protocolo MQTT

No ano de 1999, os engenheiros Andy Stanford-Clark (IBM) e Arlen Nipper (Cirrus Link, Eurotech), desenvolveram a primeira versão do protocolo MQTT (*Message Queuing Telemetry Transport*). Este tem como objetivo de proporcionar um baixo consumo de rede e banda, assim como, pretende realizar comunicação sem necessidade de muitos recursos de *software*. Apesar de o desenvolvimento do protocolo ter ocorrido no final da década de 90, a sua liberação gratuita foi disponibilizada apenas em 2010 [41].

O protocolo MQTT, ou transporte de telemetria de enfileiramento de mensagens, é um protocolo de transporte de mensagens no formato Cliente/Servidor. Este possibilita a comunicação M2M (*Machine to Machine*) e vem sendo amplamente utilizado para conectividade de sistemas com soluções IoT (*Internet of Things*).

2.4.2 Características e Formas de Comunicação

Em um protocolo MQTT, o envio e a recepção de mensagens são realizados através de um servidor intermediário chamado *Broker*. Sua forma de comunicação segue uma norma de publicação e assinante (*Publisher / Subscriber*), onde os clientes podem ser *Publishers* e/ou *Subscribers*. Quando atua como *Publisher*, o cliente envia dados para o *Broker* em um tópico específico, já quando atua como *Subscriber*, se inscreve para receber os dados de um ou mais tópicos [42]. Uma exemplificação pode ser observada na Figura 2.11.

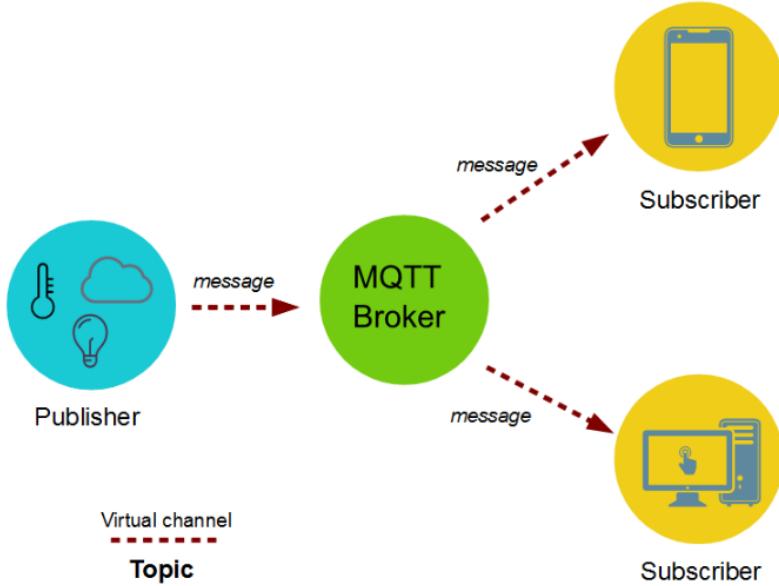


Figura 2.11: Exemplo do protocolo MQTT [7].

De uma forma mais organizada, a comunicação MQTT pode ser dividida entre [41]:

- Servidor (*Broker*): Intermediário encarregado por fazer a ponte entre o *Publisher* e o *Subscriber*, responsável por realizar a recepção, enfileiramento e envio das mensagens;
- Cliente (*Client*): Elemento que tem a capacidade de interagir com o *Broker*, tem a capacidade de enviar e receber mensagens;
- Mensagem (*Message*): Pacote de dados trocado entre cliente e servidor;
- *Payload*: Conteúdo da mensagem;
- Tópico (*Topic*): Endereço pelo qual as mensagens serão enviadas;
- *Publisher*: Dispositivo responsável pelo envio de mensagens em um determinado tópico;
- *Subscriber*: Dispositivo assinante de um tópico responsável por receber as mensagens.

2.4.3 Eclipse Mosquitto

De acordo com o site oficial [43], o Eclipse Mosquitto é um *Broker* de mensagens de código aberto que implementa o protocolo MQTT. Este *Broker* permite uma troca de mensagens leves e adequadas para diversos tipos de dispositivos, desde simples controladores de baixo consumo até servidores complexos.

Além disso, o projeto Eclipse Mosquitto provém uma biblioteca C para a implementação de clientes MQTT, assim como, alguns comandos básicos pré-estabelecidos como *mosquitto pub* e *mosquitto sub* para rápida e fácil implementação de clientes MQTT.

2.5 Modelos de Previsão para Demanda Energética

2.5.1 Demanda Energética e Curva de Carga

A demanda energética pode ser definida como o consumo energético de todos os consumidores em um determinado setor ou uma região geográfica. Dessa forma, a curva de carga pode ser definida como uma função temporal da demanda energética [44]. Sendo assim, uma análise sobre as curvas de carga expressa os padrões de necessidades energéticas conforme o período do dia, a época do ano e as diferentes regiões.

As Figuras 2.12, 2.13 e 2.14 exemplificam uma curva de carga do setor residencial dos últimos 13 anos, uma curva de carga mensal do setor residencial e uma curva de carga horária do sistema elétrico, respectivamente.

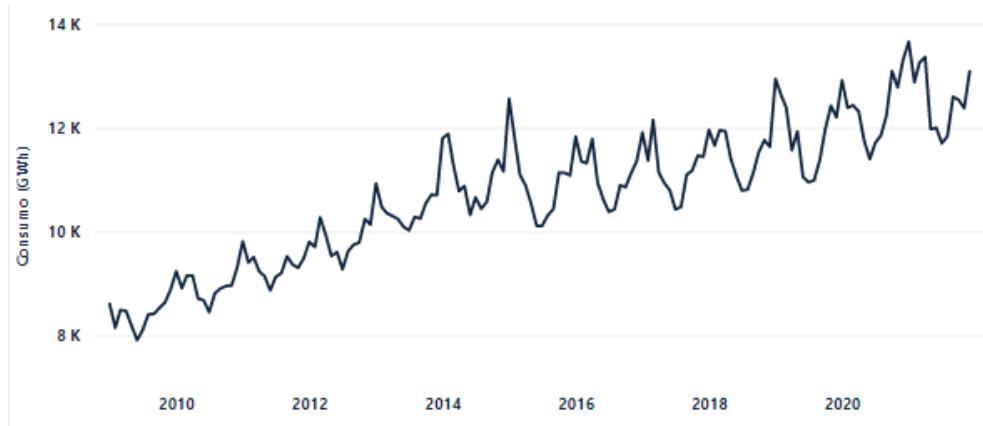


Figura 2.12: Curva de Carga Residencial do Brasil entre 2008 e 2021 [8].

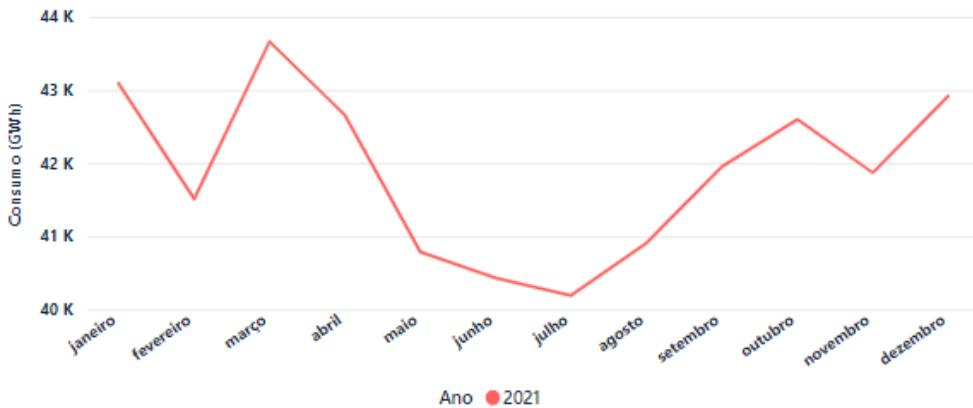


Figura 2.13: Curva de Carga Mensal Residencial do Brasil em 2021 [8].

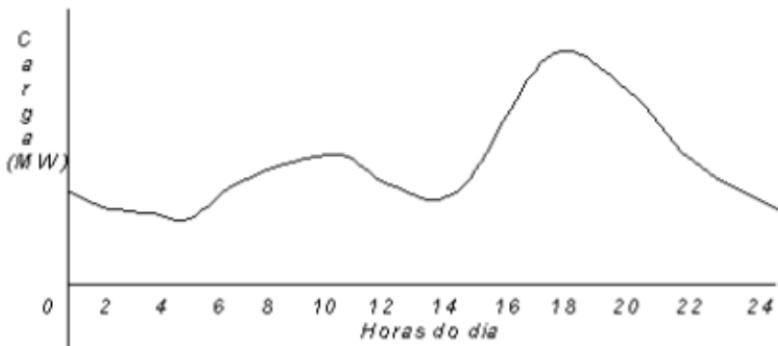


Figura 2.14: Curva de Carga Horária Típica do Sistema Elétrico [9].

Segundo Luiz Oliveira [44], é importante ressaltar que toda a energia gerada deve ser consumida imediatamente, isto porque as possibilidades de armazenamento de energia elétrica, no momento, são economicamente inviáveis. Dessa forma, do ponto de vista de planeamento é importante prever o consumo ou gerar cenários confiáveis de consumo futuro e então antecipar ações que permitam o fornecimento de energia de forma estável.

2.5.2 Inteligência Artificial

Com a finalidade de realizar previsões de consumo futuros e antecipações de ações, torna-se interessante a utilização de algoritmos de inteligência artificial. De uma forma simplificada, inteligência artificial refere-se a sistemas ou máquinas que imitam a inteligência humana, buscam padrões de funcionamento para

realizar tarefas complexas e tem a capacidade de se aprimorar iterativamente com base nas informações coletadas.

Entre os inúmeros possíveis formatos de análise de dados, diversos algoritmos de inteligência artificial podem ser propostos para realizar previsões sobre os resultados futuros. Dentre esses algoritmos destaca-se as análises de séries temporais, estas lidam com métodos estatísticos para analisar e modelar uma sequência ordenada de observações. A variável tempo é normalmente utilizada para a ordenação das observações, principalmente em casos de valores com intervalos regulares [45].

Devido ao facto de a curva de carga ser uma função temporal e intuitivamente possuir características de sazonalidades e tendência, os modelos de séries temporais revelam-se adequados para a sua análise. Dentre os modelos possíveis para a análise de séries temporais destaca-se o uso de redes neurais artificiais (RNA)"

As RNA são métodos de *Deep Learning* com capacidade de reconhecimento de padrões complexos ou numerosos e com função de aprendizado dentro da própria rede. Estas baseiam-se na arquitetura dos neurônios humanos e destinam-se a reproduzir o aprendizado por meio do desenvolvimento de sistemas que aprendem com exemplos de treino [46].

As RNA possuem diversos subconjuntos de modelos, dentre esses, o modelo de rede neuronal recorrente (RNR) apresenta eficiência para aprendizado com dados sequenciais e temporais. Modelos RNR incluem um estado oculto, que é reciclado para produzir o próximo modificado, e seu funcionamento se assemelha ao das memórias humanas, que reciclam consciência de estados anteriores para interpretar corretamente novos dados [46].

As RNR simples, sofrem com o problema de gradiente de fuga, o que significa que a rede só pode se lembrar de entradas recentes e rapidamente esquece as entradas de longo prazo. Para lidar com esse problema, foi introduzida uma variante do RNR conhecida como redes de memória de longo prazo, ou LSTM. Os LSTM alcançam capacidade de memória de longo prazo e tornaram-se uma variante convencional para uso de algoritmos de RNR [47].

Capítulo 3

Hardware

Este capítulo pretende esquematizar o sistema completo de *smart grid* com comunicação LoRa. Além disso, visa também detalhar os protótipos construídos: *smart meter* e *gateway*, assim como, todas as ligações e componentes necessários para prototipagem.

3.1 Esquemático do Sistema

Na simulação de uma rede *smart grid* utilizando protocolo de comunicação LoRaWAN faz-se necessário: a medição dos valores de tensão e corrente da rede elétrica (*smart meter*), a transmissão destas medições para um receptor conectado à internet (*gateway*), a serialização e transferências desses dados para um servidor MQTT, o armazenamento adequado destas medições em um servidor de banco de dados e uma plataforma que permita a fácil visualização destes dados (*dashboard*). O esquemático completo do sistema pode ser observado na Figura 3.1.

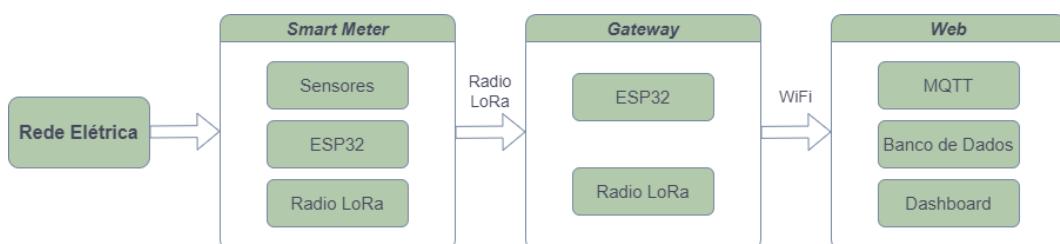


Figura 3.1: Esquemático do Sistema.

3.2 Smart Meter

3.2.1 Esquemático do Smart Meter

Para a prototipagem do dispositivo *smart meter* foi selecionado os seguintes componentes:

- ESP32 DOIT Devkit v1: A placa de desenvolvimento ESP32 selecionada para microcontrolador do sistema;
- LoRa E32-433T20D: Rádio para transmissão dos dados;
- Display LCD 16x2: Tela para a visualização dos dados de tensão e corrente instantâneas;
- ZMPT101B: Sensor para a medição da tensão elétrica da rede;
- ACS712T: Sensor para a medição da corrente elétrica da rede.

O esquemático do dispositivo *smart meter* pode ser observado na Figura 3.2.

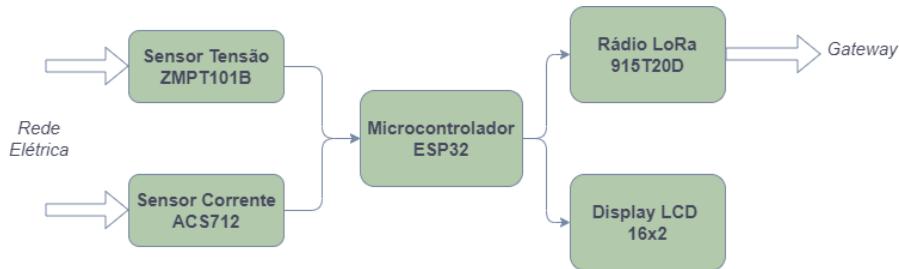


Figura 3.2: Esquemático do *Smart Meter*.

A imagem do protótipo do *smart meter* pode ser observada na Figura 3.3.

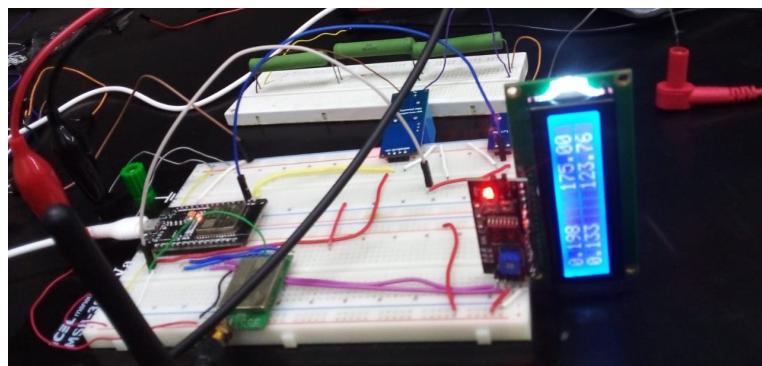


Figura 3.3: Imagem do Protótipo *Smart Meter*.

3.2.2 Sensor Tensão - ZMPT101B

Para a realização das medições de tensão na rede elétrica, foi selecionado o sensor ZMPT101B. Este sensor tem a capacidade de operar em corrente alterna entre tensões de 0 V a 1000 V, sendo ideal para a realização de medições de residências monofásicas e bifásicas [48]. A imagem do componente pode ser observada na Figura 3.4

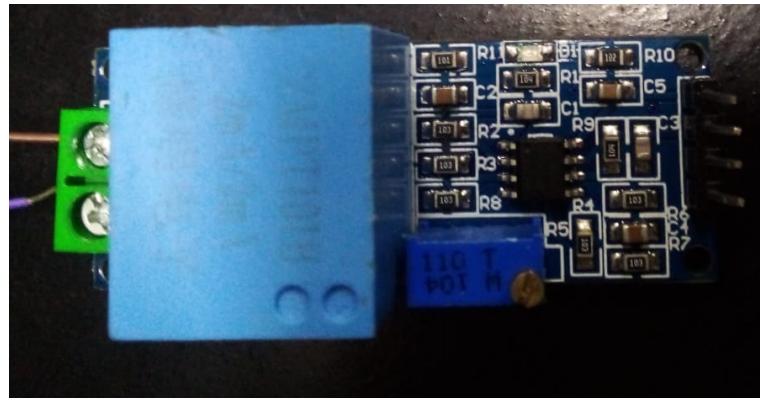


Figura 3.4: Sensor de Tensão ZMPT101B [10].

O sensor ZMPT101B possui seis pinos de conexões: dois pinos para a conexão de fase e neutro da rede elétrica, um pino de VCC que deve ser alimentado com tensões entre 3,3 V e 5 V, dois pinos de *ground* e um pino de sinal. A imagem do *pinout* do sensor pode ser observada na Figura 3.5.



Figura 3.5: Pinout do Sensor de Tensão ZMPT101B [10].

O funcionamento do sensor ZMPT101B segue uma transformação de tensão para tensão, porém de uma tensão mais alta sem valores de corrente contínua, no caso a rede elétrica, para uma tensão mais baixa com um valor de corrente contínua de metade da alimentação do VCC. Em outras palavras, supondo que

o sensor seja alimentado com uma tensão de 5 V, então, no instante que a tensão da rede elétrica estiver em 0 V o sinal do sensor estará com um valor 2,5 V. Logo, se a alimentação do sensor estiver utilizando um valor de tensão de 3,3 V, no instante que a tensão da rede elétrica estiver em 0 V o sinal do sensor estará marcando 1,65 V.

No caso do protótipo desenvolvido, foi utilizado uma tensão de alimentação de 3,3 V. Dessa forma, em instantes que a rede elétrica apresentar uma tensão superior a 0 V, o pino de sinal do sensor deve apresentar uma tensão entre 1,65 V e 3,3 V. Em momentos em que a rede elétrica apresente tensões negativas, ou seja, inferiores a 0 V, o pino de sinal do sensor deve apresentar valores de tensão entre 0 V a 1,65 V. Um potenciômetro externo é responsável por realizar os ajustes desses valores. Para o protótipo desenvolvido, devido ao intuito de medições residenciais, foi utilizado a relação de uma tensão de 180 V positivos na rede equivalendo a um sinal de 2,08 V e uma tensão de 180 V negativos na rede equivalendo a um sinal de 1,28 V. Sendo assim, todos os valores do intervalo aproximado entre -750 V e 750 V da rede elétrica seguem uma relação linear entre os valores de 0 V e 3,3 V do sinal de saída do sensor. Dessa forma, com esse largo intervalo de medição, o sistema permite facilmente a aquisição de dados de residências monofásicas e bifásicas.

3.2.3 Sensor Corrente - ACS712T

Para a realização das medições de corrente da rede elétrica, foi selecionado o sensor de ACS712T. Este sensor funciona de forma invasiva, ou seja, ele necessita entrar em série junto a carga que se deseja medir os valores de corrente. A imagem do sensor pode ser observada na Figura 3.6.

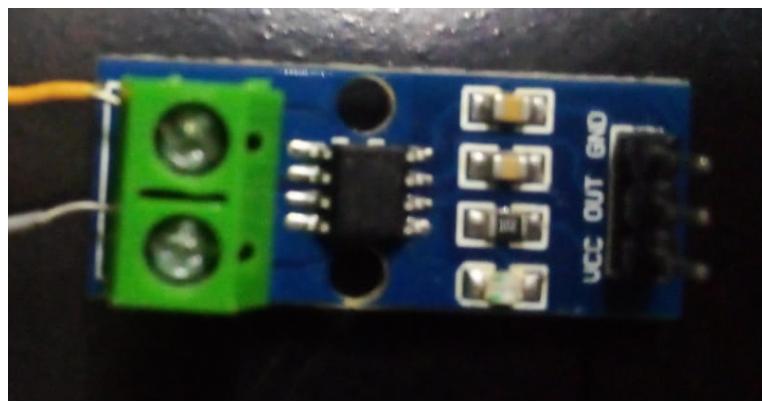


Figura 3.6: Sensor de Corrente ACS712T.

Existem três modelos do sensor ACS712T: ELC-05B, ELC-20A e ELC-30A. Esses modelos diferenciam-se em apenas em relação à sua capacidade máxima de

corrente suportada e sua precisão de medição, sendo suas correntes máximas 5 A, 20 A e 30 A respectivamente. Entretanto, quanto menor a corrente suportada maior a precisão do medidor. Os valores de precisão para cada modelo do sensor podem ser observados na Figura 3.7.

Part Number	Packing*	T_A (°C)	Optimized Range, I_P (A)	Sensitivity, Sens (Typ) (mV/A)
ACS712ELCTR-05B-T	Tape and reel, 3000 pieces/reel	-40 to 85	±5	185
ACS712ELCTR-20A-T	Tape and reel, 3000 pieces/reel	-40 to 85	±20	100
ACS712ELCTR-30A-T	Tape and reel, 3000 pieces/reel	-40 to 85	±30	66

Figura 3.7: Tabela de Precisão do Sensor ACS712T [11].

O sensor ACS712T possui cinco pinos de conexão: dois pinos para a conexão de fase e neutro da rede elétrica, um pino de VCC que deve ser alimentado com uma tensão de 5 V, um pino de ground e um pino de sinal. A imagem do *pinout* do sensor pode ser observada na Figura 3.8.

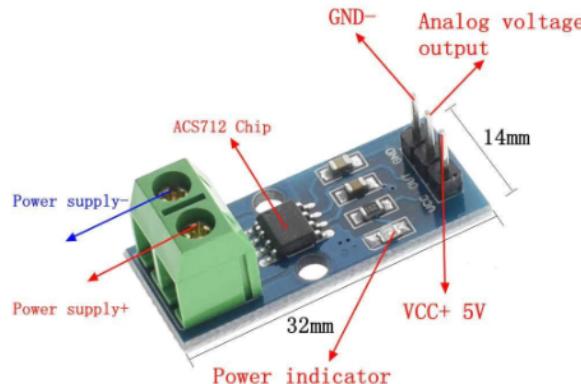


Figura 3.8: Pinout do Sensor ACS712T [12].

Assim como no caso do sensor de tensão de ZMPT101B, o sensor de corrente ACS712T também utiliza um *offset* de metade da tensão de VCC no seu pino de sinal. Para o caso do protótipo, foi selecionado o sensor de ACS712T ELC-05B, em que a relação entre a corrente da rede elétrica e o sinal de saída segue uma relação de 185 mV/A, como pode ser observado na Figura 3.8.

Dessa forma, supondo que em um determinado momento o circuito esteja marcando um valor de corrente de 2 A, neste caso, o pino de sinal do sensor ACS712T deve marcar um valor de 2,87 V de tensão. Da mesma forma, se em

um determinado momento o circuito esteja marcando um valor de corrente de -1 A, neste caso, o pino de sinal do sensor deve marcar um valor de 2,315 V.

3.2.4 Microcontrolador *Smart Meter* - ESP32

Para a aquisição, processamento e cálculo dos valores de medição foi selecionado a placa de desenvolvimento ESP32 Devkit v1, a qual possui como microcontrolador o ESP32 da fabricante espressif. A Figura 3.9. ilustra a placa ESP32 Devkit v1.



Figura 3.9: Placa ESP32 Devkit v1.

O microcontrolador ESP32 possui: 40 MHz de frequência de *clock* máxima, 32 bits de barramento, 448 KB de memória ROM para *boot* e funções essenciais, 520 KB de SRAM para instruções e dados, 34 GPIO, 2 conversores analógico digital de 12 bits com suporte em 18 canais, suporte para conexão Wifi e Bluetooth [49]. O *pinout* da placa de desenvolvimento ESP32 Devkit v1 pode ser observado na Figura 3.10.

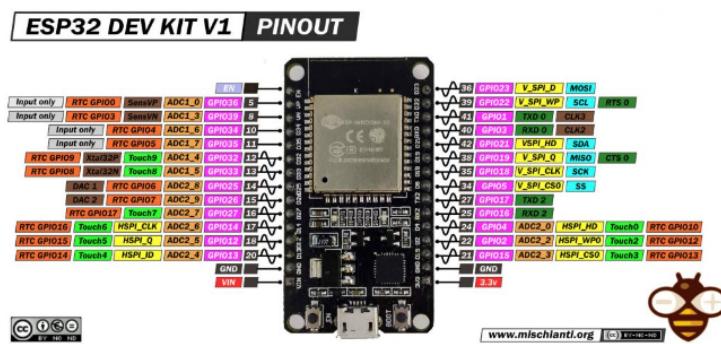


Figura 3.10: Pinout da Placa de Desenvolvimento ESP32 Devkit v1 [13].

Para o projeto do *smart meter* foi utilizado dois conversores analógicos digitais do ESP32 para realizar a conversão dos valores coletados pelos sensores ZMPT101B e ACS712T. Após a coleta dos dados de medição, são realizados cálculos para estabelecer a frequência da rede elétrica. Posteriormente, os valores de tensão e corrente são impressos no *display LCD* para a visualização do consumidor. Por fim, os dados medidos são enviados em intervalos regulares para a placa LoRa 433T20D que se encarrega de realizar a transmissão dos dados.

3.2.5 Rádio LoRa *Smart Meter* - 433T20D

Para a transmissão dos dados coletados pelo *smart meter*, foi utilizado a placa de rádio LoRa E32-433T20D. Esta placa da Ebyte possui o *chip* SX1278 da Semtech Corporation, 433 MHz de frequência, 10 a 20 dbm de potência e interface de comunicação UART. A imagem da placa pode ser observada na Figura 3.11.

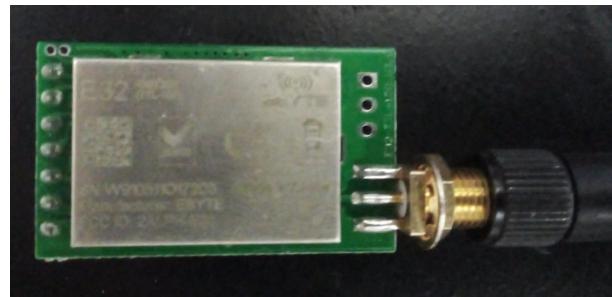


Figura 3.11: Placa LoRa E32-433T20D.

O *pinout* da placa LoRa E32-433T20D para realização da conexão com o ESP32 pode ser observado na Figura 3.12.



Figura 3.12: *Pinout* da Placa LoRa E32-433T20D [14].

No projeto, a placa LoRa foi utilizada para realizar a transmissão dos dados processados pelo microcontrolador ESP32 para o *gateway*. Para isso, a placa LoRa é alimentada com uma tensão de 3,3 V fornecida pelo ESP32. Para comunicação entre a placa LoRa e o ESP32 é realizado a ligação das entradas TX e RX da placa com as entradas série TX e RX do ESP32. Os pinos restantes são conectados ao *ground* do sistema.

3.3 *Gateway*

3.3.1 Esquemático do *Gateway*

Para a prototipagem do dispositivo *gateway* foi selecionado os seguintes componentes:

- ESP32 DOIT Devkit v1: A placa de desenvolvimento ESP32 selecionada para microcontrolador do sistema;
- LoRa E32-433T20D: Rádio para recepção dos dados.

O esquemático completo do dispositivo pode ser observado na Figura 3.13.



Figura 3.13: Esquemático do *Gateway*.

A imagem do protótipo do dispositivo pode ser observada na Figura 3.14.

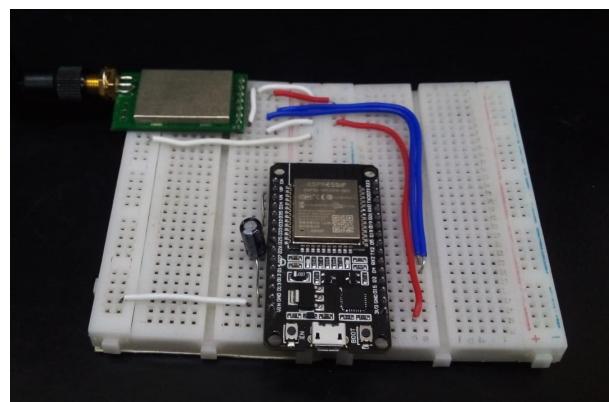


Figura 3.14: Imagem do Protótipo *Gateway*.

3.3.2 Microcontrolador *Gateway* - ESP32

Para o armazenamento e encaminhamento para *web* dos valores de medição enviados do *smart meter* para o *gateway*, foi igualmente selecionado a placa de desenvolvimento ESP32 Devkit v1. Assim como mencionado, esta placa possui como microcontrolador o ESP32 da fabricante espressif. A Figura 3.9 ilustra a imagem da placa ESP32 Devkit v1.

No caso do *gateway*, a utilização do microcontrolador ESP32 facilita a implementação do protótipo, como dito, este microcontrolador possui um módulo interno que permite conexões Wifi. Dessa forma, permite a comunicação *web* para envio dos dados ao servidor MQTT sem a necessidade de um módulo externo para conexões Wifi ou Ethernet.

3.3.3 Rádio LoRa *Gateway* - 433T20D

Assim como para o *smart meter*, a placa LoRa 433T20D foi selecionado para o *gateway*. Assim como mencionado, está placa da Ebyte possui o *chip* SX1278 da Semtech Corporation, 433 MHz de frequência e interface de comunicação UART. A imagem da placa pode ser observada na Figura 3.11.

Para o caso do *gateway*, a placa LoRa 433T20D tem a funcionalidade de receber os dados enviados pelo rádio LoRa do *smart meter* e encaminhar para o microcontrolador ESP32, que por sua vez, se responsabiliza de realizar o encaminhamento para a *web*.

Capítulo 4

Software

Este capítulo pretende apresentar os *softwares* necessários para construção do protótipo de *smart grid* com comunicação LoRa. Além disso, visa também utilizar fluxogramas para detalhar o funcionamento dos códigos utilizados para a configuração do *smart meter*, *gateway*, servidor MQTT, servidor de banco de dados, *dashboard* para visualização de dados e algoritmos para previsão de demanda do consumidor.

4.1 Código *Smart Meter*

4.1.1 Fluxograma - Cálculo da Tensão

O algoritmo de cálculo da tensão implementado no *software* do *smart meter* pode ser observado na Figura 4.1. Inicialmente, são declaradas as variáveis inteiiras pinTensao, i, tensaoInst e maiorValor. A seguir, é atribuído o valor zero para as variáveis maiorValor e i. Posteriormente, são declaradas as variáveis reais Vp e tensaoRMS. Após a definição das variáveis, é iniciado o *loop* de funcionamento do microcontrolador ESP32, sendo representado pela condição *while* “1 == 1” no fluxograma.

No início do *loop* de funcionamento é realizado a leitura da variável pinTensao, esta recebe o valor da porta GPIO física do ESP32, responsável por ser ligada à saída do sinal do sensor de tensão ZMPT101B. Posteriormente, a variável tensaoInst recebe a leitura do pinTensao após a conversão ADC (análogo-digital) realizada pelo ESP32.

Após a leitura da tensão instantânea é verificado a condição do contador i, a qual é responsável por definir o número de leituras realizas em um certo intervalo, neste caso a cada 30000 leituras. Caso a variável i seja menor de 30000 é

verificado em uma nova condição o caso de a variável tensaoInst ser maior que o maiorValor, neste caso, a variável maiorValor recebe o valor da variável tensaoInst, caso contrário, nada acontece. Dessa forma, a variável maiorValor vai receber ao final de todas as iterações o maior valor lido pela a variável tensaoInst. Caso a variável i seja igual ou maior que 30000, então, é realizado a conversão do maiorValor para a variável Vp, o cálculo da tensão RMS é efetuado pela divisão do valor de pico por raiz de dois, em seguida, é realizado a reinicialização das variáveis i e maiorValor para zero. O código completo do *smart meter* pode ser observado no Apêndice A.

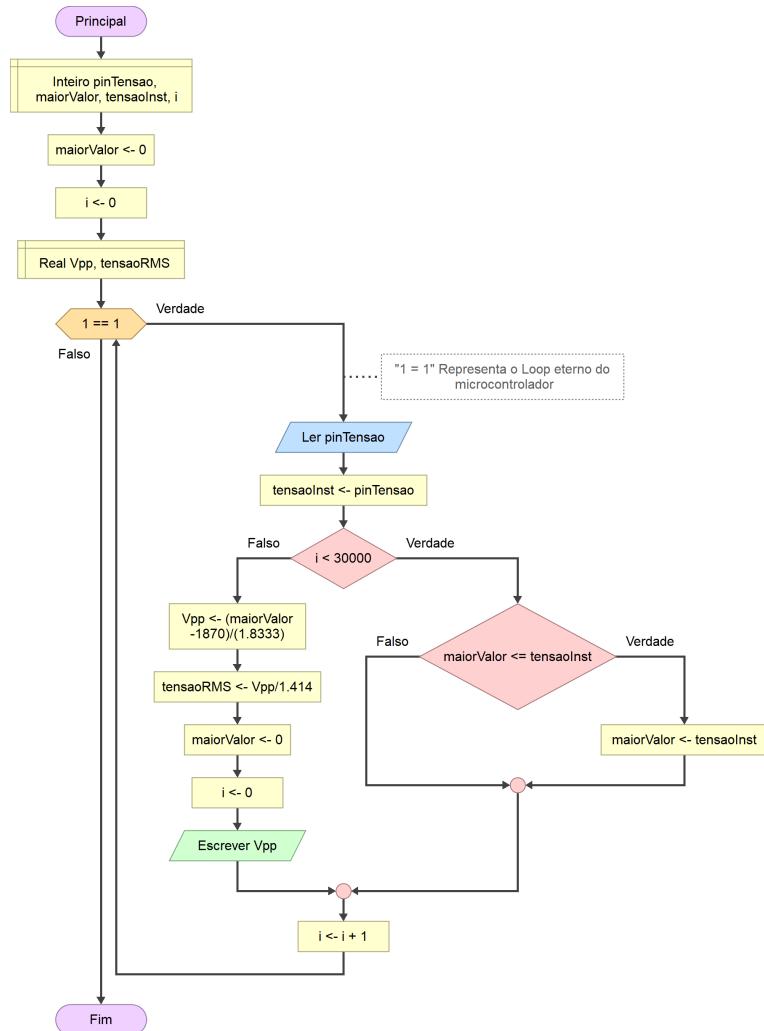


Figura 4.1: Fluxograma Código do Cálculo de Tensão do *Smart Meter*.

A fórmula de cálculo utilizada para a conversão entre o valor lido após a conversão ADC e o valor de tensão real na rede elétrica pode ser observado na expressão (4.1). O valor de 1870 como 0 V e o valor de 2200 como 180 V foram adquiridos experimentalmente com a realização de testes práticos.

$$Tensao = ((ValorDigital - 1870) \cdot (180 - 0)) / (2200 - 1870) \quad (4.1)$$

4.1.2 Fluxograma - Cálculo da Corrente

O algoritmo de cálculo da corrente implementado no *software* do *smart meter* tem muitas semelhanças ao algoritmo de cálculo da tensão, dessa forma, não é necessária uma segunda explicação de seu fluxograma. A imagem do fluxograma do cálculo de corrente pode ser observada na Figura 4.2. O código completo do *smart meter* pode ser observado no Apêndice A.

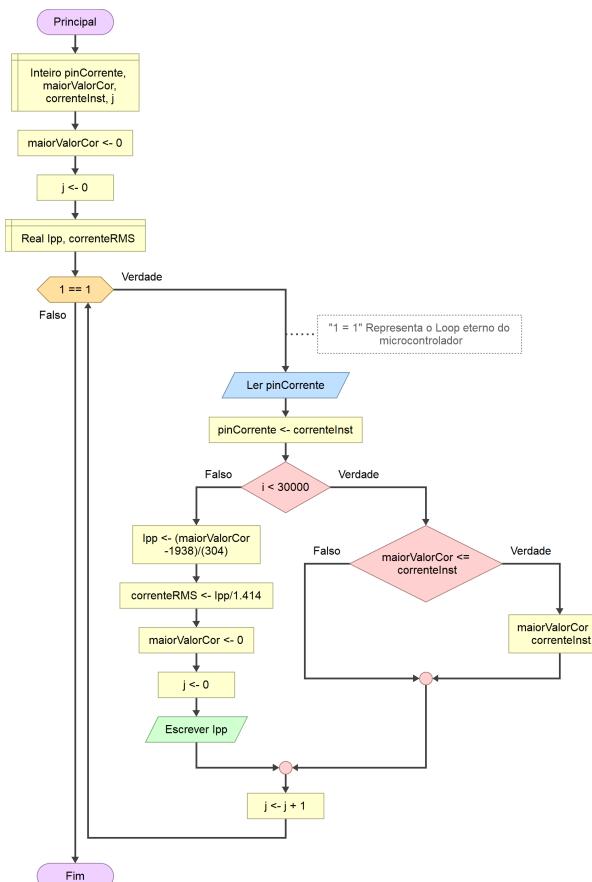


Figura 4.2: Fluxograma Código do Cálculo da Corrente do *Smart Meter*.

A fórmula de cálculo utilizada para a conversão entre o valor lido após a conversão analógico digital e o valor de corrente real na rede elétrica pode ser observado na expressão (4.2).

$$\text{Corrente}[A] = ((\text{ValorDigital} - 1938) \cdot (5 - 0)) / (2655 - 1938) \quad (4.2)$$

O valor de 1938 como 0 V foi obtido experimentalmente com a realização de testes práticos. A relação de equivalência entre 5 A e o valor de 2655 lido após a conversão ADC foi obtida utilizando a tabela de conversão que pode ser observada na Figura 3.7. O cálculo detalhado da conversão pode ser observado nas expressões (4.3) e (4.4).

$$5[A] = 2,5 + (0,185 \cdot 5)[V] = 3,425[V] \quad (4.3)$$

$$2655 = (1938 \cdot 3,425) / 2,5 \quad (4.4)$$

4.1.3 Fluxograma - Cálculo da Frequência e Envio de Dados LoRa

O algoritmo de cálculo da frequência da rede e envio de dados LoRa implementado no *software* do *smart meter* pode ser observado na Figura 4.3. O código completo do *smart meter* pode ser observado no Apêndice A.

Inicialmente, são declaradas as variáveis inteiros k, pinTensao e tensaoInst. A seguir, é atribuído o valor zero para a variável k. Posteriormente, é declarado a variável real freq. Após a definição das variáveis, é iniciado o *loop* de funcionamento do microcontrolador ESP32, sendo representado pela condição *while* “1 == 1” no fluxograma.

Assim como apresentado no algoritmo de cálculo de tensão, o *loop* de funcionamento realiza a leitura da variável pinTensao, que recebe o valor da porta GPIO física do ESP32 responsável por ser ligada a saída do sinal do sensor de tensão ZMPT101B. Posteriormente, a variável tensaoInst recebe a leitura do pinTensao após a conversão ADC (análogo-digital) realizada pelo ESP32.

A seguir, é verificada a condição da variável tensaoInst, caso a variável apresente algum valor entre 1868 e 1872 é incrementado 1 ao valor do contador k. O contador k realiza a contagem de quantas vezes que a variável ultrapassa o eixo x, representando o 0 V da rede elétrica. Dessa forma, como o sinal é sinusoidal, o número de vezes que o valor da rede elétrica chega a 0 V dividido pelo tempo em segundos, representa a frequência da rede em Hertz.

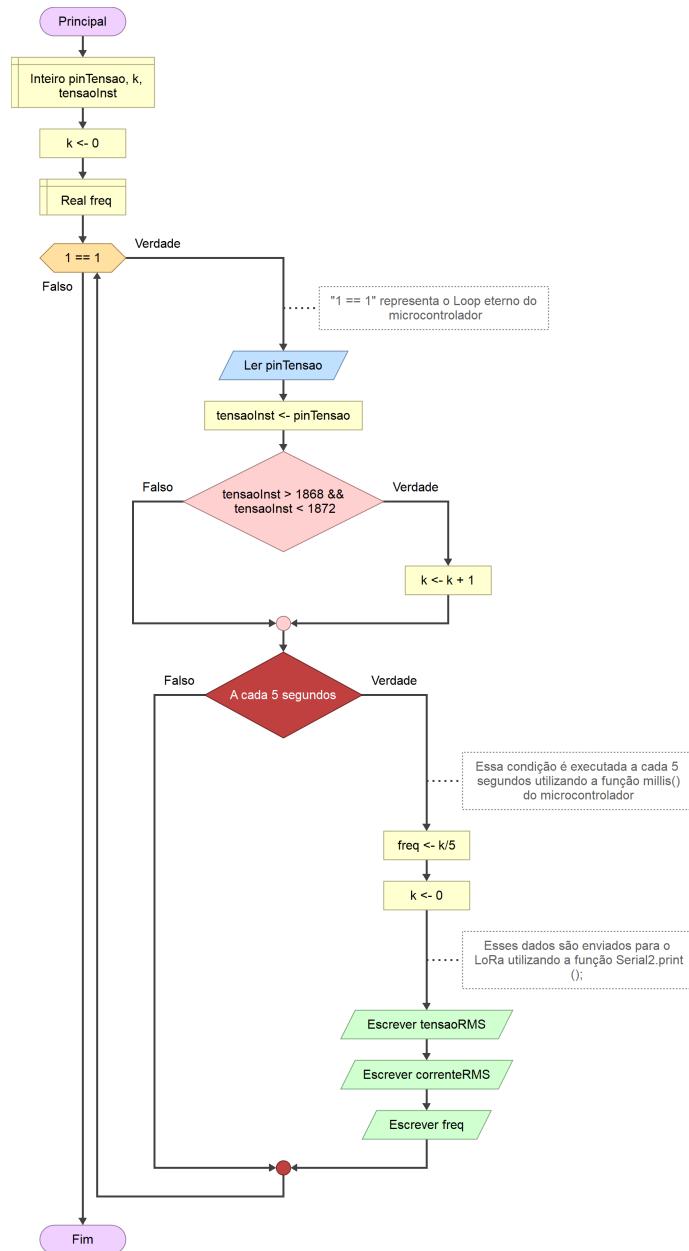


Figura 4.3: Fluxograma Código do Cálculo da Frequência e Envio de Dados LoRa do *Smart Meter*.

O valor de 0 V da rede elétrica foi estabelecido como 1870, porém, devido ao intervalo de amostragem, eventualmente não é possível verificar exatamente o valor 1870. Devido a isso, o intervalo entre 1868 e 1872 foi estabelecido realizando testes experimentais.

Após a coleta dos dados de frequência, é iniciado o processo de envio dos

dados para o *gateway* via rádio LoRa. Para tal objetivo, é estabelecido uma condição que se torna verdadeiro a cada 5 segundos. Para a contagem de tempo é utilizado a função `millis()` da biblioteca padrão. Dessa forma, a cada 5 segundos é enviado um pacote contendo o valor de tensão RMS, corrente RMS e frequência da rede para o *gateway*.

4.2 Código *Gateway*

4.2.1 Fluxograma - Conexão Wifi e Recepção dos Dados via LoRa

O algoritmo implementado no *software* do *gateway* responsável pela conexão Wifi e a recepção dos dados enviados pelo *smart meter* pode ser observado na Figura 4.4. O código completo do *gateway* pode ser observado no Apêndice B.

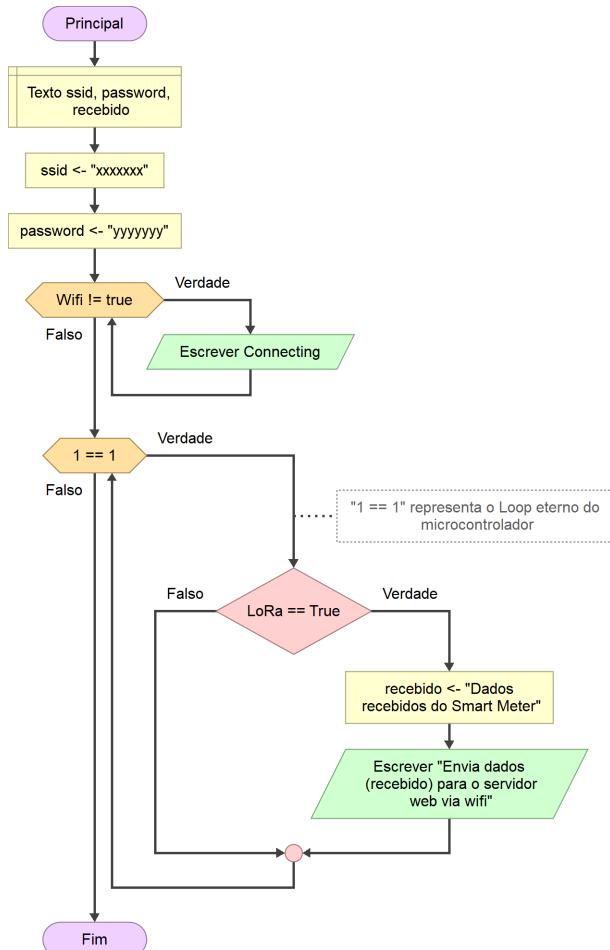


Figura 4.4: Fluxograma Código Conexão Wifi e Recepção via LoRa do *Gateway*.

Inicialmente, são declaradas as variáveis de texto ssid, password e received. A seguir, são atribuídos os valores da rede Wifi local nas variáveis ssid e password. Posteriormente, é executado a função que tenta realizar a conexão com a rede local. Após o sucesso da conexão é iniciado o *loop* de funcionamento do microcontrolador ESP32, sendo representado pela condição while “*1 == 1*” no fluxograma.

Dentro do *loop* de funcionamento, é executado uma função que verifica a recepção de dados no rádio LoRa, caso ocorra a recepção de algum pacote, os dados são serializados e enviados para o servidor *web* via Wifi.

4.3 Código MQTT

4.3.1 Descrição - Conexão com o Servidor MQTT

Para realizar a conexão dos dados enviados pelo *gateway* para o servidor MQTT foi utilizado um algoritmo em Python, o qual pode ser executado a um determinado intervalo de tempo regular. O código completo do algoritmo de conexão com o servidor MQTT pode ser analisado no Apêndice C.

Inicialmente, o algoritmo realiza uma requisição ao servidor *web* roteado pelo *gateway* para coleta dos dados de medição. Após a requisição, os dados recebidos são organizados e serializados. Este trecho do código pode ser observado na Figura 4.5.

```
x = 0
while(x == 0):
    try:
        # Recebe o body da pagina web
        response = req.get(url, headers=headers)
        # Divide o body do html em uma string por espaçoamento.
        parametro = response.text.split()
        print(response.text.split())
        x = 1
    except:
        x = 0
```

Figura 4.5: *Loop* de Requisição ao Servidor Web Roteado pelo *Gateway*.

Após a serialização dos dados, é atribuído os valores de data e hora ao pacote. Posteriormente, com o pacote completo e organizado, é iniciado a conexão e envio destes dados para o servidor MQTT no tópico smartmeter1. Este trecho do código pode ser observado na Figura 4.6.

```

# Adiciona data e hora no pacote
horario = str(datetime.datetime.now())
parametro.append(horario)

# Realiza a publicação no tópico "smartmeter1".
client1.publish(
    "smartmeter1",
    parametro[0] + " " +
    parametro[1] + " " +
    parametro[2] + " " +
    parametro[3]
)

```

Figura 4.6: Adição de Data/Hora e Envio do Pacote ao Servidor MQTT.

4.4 Código Banco de Dados

4.4.1 Descrição - Conexão com o Servidor de Banco de Dados

Para realizar o armazenamento dos valores do servidor MQTT em um banco de dados foi utilizado um algoritmo em Python, programado para se manter em um *loop* eterno à espera de publicações nos tópicos inscritos. O código completo do algoritmo de armazenamento dos dados em um servidor de banco de dados pode ser analisado no Apêndice D.

Dentro do *loop* de funcionamento é definido os parâmetros necessários para realizar a conexão ao *broker* MQTT: endereço IP do *broker*, porta de conexão, nome do cliente e tópico para subscrição. Além disso, também é atribuído os nomes para as funções de *callback* utilizadas na recepção das mensagens. Este trecho do código pode ser observado na Figura 4.7.

```

broker_address="localhost"
print("creating new instance")
client = paho.Client("medidor")
print("connecting to broker")
client.connect(broker_address, 1884)
print("Subscribing to topic","smartmeter1")
client.on_message = on_message
client.subscribe("smartmeter1",1)

# Manter o script sempre ligado para caso de publicacoes
client.loop_forever()

```

Figura 4.7: Loop de Funcionamento e Parâmetros de Conexão.

A seguir, na Figura 4.8. pode ser observado a função de *callback* para novas publicações no tópico MQTT. Esta função é responsável por imprimir os valores de qualidade da conexão e encaminhar o pacote para a função de armazenamento no servidor de banco de dados.

```
def on_message(client, userdata, message):
    print("message received " ,str(message.payload.decode("utf-8")))
    print("message topic=",message.topic)
    print("message qos=",message.qos)
    print("message retain flag=",message.retain)
    salvaBD(str(message.payload.decode("utf-8")))
```

Figura 4.8: Função de *Callback* para Publicações no Tópico MQTT.

A função de armazenamento dos dados recebidos no banco de dados pode ser observada na Figura 4.9. Esta função é responsável por serializar os dados recebido pelo pacote MQTT para o formato aceite pelo banco de dados e a seguir realizar seu registo.

```
def salvaBD(mensagem):
    m = mensagem.split()
    lista = [
        (m[3] + " " + m[4], m[0], m[1], m[2]),
    ]
    cur.executemany("insert into registros values (?, ?, ?, ?)", lista)
    con.commit()
```

Figura 4.9: Função de Amazenamento de Dados no Banco de Dados.

4.5 Ligação com *Dashboard*

4.5.1 Descrição - Conexão entre Banco de Dados e *Dashboard*

Para permitir a visualização dos dados armazenados no servidor de banco de dados, foi utilizado um *script* em Python junto da biblioteca de construção de gráficos matplotlib. O código completo do *script* pode ser analisado no Apêndice E.

Inicialmente, o algoritmo realiza uma chamada ao banco de dados para a organização dos dados nas devidas variáveis de manipulação. Esse trecho do código pode ser observado na Figura 4.10.

Posteriormente, é realizado algumas conversões de tipos de variáveis e padronizações dos dados de data e hora. Dessa forma, com os dados organizados, é definido os parâmetros do gráfico, como o título e os rótulos das variáveis, e por fim, é realizada a visualização do gráfico construído. Esse trecho do código pode ser observado na Figura 4.11.

```
con = sqlite3.connect('bancoDeDados.db')
cur = con.cursor()

sqlite_select_query = """SELECT * from registros"""
cur.execute(sqlite_select_query)
records = cur.fetchall()

data.append(row[0])
tensao.append(row[1])
corrente.append(row[2])
frequencia.append(row[3])
```

Figura 4.10: Código de Consulta ao Banco de Dados.

```
plt.figure(1)          plt.figure(2)
plt.title("Corrente X Horário")  plt.title("Tensão X Horário")
plt.xlabel("Horário")        plt.xlabel("Horário")
plt.ylabel("Corrente [A]")    plt.ylabel("Tensão [V]")
plt.grid()                plt.grid()
plt.plot(datasF, correnteF) plt.plot(datasF, tensaoF)
plt.show()
```

Figura 4.11: Código para Visualização dos Dados .

4.6 Descrição - Algoritmo de Modelo de Previsão de Carga

4.6.1 Tratamento do Banco de Dados

Para permitir uma análise estatísticas e a utilização de algoritmos de previsão de demanda, o banco de dados foi populado com dados oficiais de consumo elétrico do estado de São Paulo (Brasil). Após a aquisição dos dados, foi necessário realizar a padronização dos dados para a adequação aos algoritmos de

inteligência artificial, para isso, foi criado um *script* em Python que pode ser analisado por completo no Apêndice F.

Inicialmente, o *script* interativo pergunta o tipo de consumo que deseja ser analisado, como pode ser observado na Figura 4.12.

```
# Definicao do tipo de consumo a ser analisado
def defineTipoConsumo():
    print("Escolha o tipo de consumo: ")
    print("1 - Residencial")
    print("2 - Comercial")
    print("3 - Industrial")
    print("4 - Total")
    tipo_consumoF = int(input("Digite o numero: "))
    return tipo_consumoF
```

Figura 4.12: Código para Definir o Tipo de Consumo.

Após a escolha do tipo de consumo é padronizado os valores de mês e ano para apenas uma coluna de data. Esse trecho de código é ilustrado na Figura 4.13.

```
# Formata a data
data = pd.date_range(start='01/15/2004', end='12/15/2021', periods=len(df))
df['data'] = data.strftime('%Y-%m')
df = df.set_index('data')
df.index.freq='MS'
```

Figura 4.13: Código para Padronizar as Datas.

Por fim, são eliminadas as colunas que não serão utilizadas pelos algoritmos de previsão e realizado a conversão para extensão csv. Essa etapa do código pode ser observada na Figura 4.14.

```
# Deleta coluna de numero de consumidores
df.drop(["numero_consumidores"], axis=1, inplace=True)
# Deleta coluna ano
df.drop(["ano"], axis=1, inplace=True)
# Deleta coluna mes
df.drop(["mes"], axis=1, inplace=True)
# Deleta coluna tipo_consumo
df.drop(["tipo_consumo"], axis=1, inplace=True)
# Deleta coluna sigla_uf
df.drop(["sigla_uf"], axis=1, inplace=True)
# Converte a tabela para .CSV
df.to_csv('Ai.csv')
```

Figura 4.14: Código para Eliminar Colunas e Converter para Extensão csv.

4.6.2 Algoritmo de Previsão de Carga

Para a previsão do consumo foi utilizado o algoritmo de inteligência artificial *Long Short Term Memory* (LSTM) para previsão de séries temporais. Este algoritmo foi implementado utilizando um *script* em Python que pode ser analisado por completo no Apêndice G.

Inicialmente, é realizado a medição da quantidade de dados com o intuito de definir a relação entre número de dados que serão utilizados para treino e para teste do algoritmo. Esse trecho de código pode ser observado na Figura 4.15.

```
# Print para determinar o tamanho da tabela e
# consequentemente o tamanho de treino/teste
print(len(df))

# Treino e Teste
train = df.iloc[:204]
test = df.iloc[204:]
```

Figura 4.15: Código para Separação dos Dados entre Treino e Teste.

Posteriormente, é realizado uma transformação dos dados para eles se encaixarem em um intervalo de 0 a 1, sendo 1 o maior valor e 0 o menor. Esse trecho do código é ilustrado na Figura 4.16.

```
# Rescreve os dados para um intervalo de 0 a 1
scaler = MinMaxScaler()
scaler.fit(train)
scaled_train = scaler.transform(train)
scaled_test = scaler.transform(test)
```

Figura 4.16: Código para Transformação dos Dados.

Após a transformação dos dados, é definido a quantidade de *inputs* que serão analisados para a saída de um *output*. No caso do *script*, foram utilizados 60 meses para a previsão do mês seguinte. Esse trecho do código é ilustrado na Figura 4.17.

Com a correta padronização dos dados, é definido o algoritmo LSTM e o modelo sequencial de séries temporais. Esse trecho do código é ilustrado na Figura 4.18.

Após o processamento do algoritmo LSTM, é realizado a transformação inversa dos dados para a visualização correta dos dados previstos. Esse trecho do código é ilustrado na Figura 4.19.

```

n_input = 60 #12
n_features = 1
generator = TimeseriesGenerator(scaled_train, scaled_train, n_input, batch_size=1)
x,y = generator[0]
print("x: {}".format(x))
print("y: {}".format(y))

```

Figura 4.17: Código para Definição dos *Inputs* e *Outputs* do Algoritmo.

```

# Define o modelo
model = Sequential()
model.add(LSTM(100, activation='relu', input_shape=(n_input, n_features)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')

```

Figura 4.18: Código para Definição do Algoritmo LSTM.

```

# Retorna os dados para os valores reais
true_predictions = scaler.inverse_transform(test_predictions)

```

Figura 4.19: Código para Transformação Inversa dos Dados.

Após a correção dos dados para os valores reais, é construído uma tabela e visualizado um gráfico de comparação entre os dados reais medidos nos últimos 12 meses e os dados previstos pelo algoritmo LSTM. Esse trecho do código é ilustrado na Figura 4.20.

```

test = test.rename_axis('data').reset_index() # Reseta o index
test['forecast'] = df2['Consumo']
test = test.set_index('data')

test.plot(figsize=[12,6])
plt.show()

```

Figura 4.20: Código para Construção da Tabela e Visualização Comparativa.

Por fim, é utilizado uma fórmula *Mean Absolute Percent Error* (MAPE) para a comparação percentual entre os valores reais e os previstos. Esse trecho do código pode ser observado na Figura 4.21.

```
# Cria a coluna com o erro percentual
test['MAPE'] = (100 * (test['consumo'] - test['forecast'])) / test['consumo']

print(test)
```

Figura 4.21: Código para Implementação da Coluna de MAPE.

Capítulo 5

Resultados e Discussões

Este capítulo visa apresentar os resultados dos testes de funcionamento, precisão e confiabilidade do sistema de *smart grid* e seus componentes. Além disso, este capítulo pretende realizar comentários e discussões a respeito dos resultados obtidos.

5.1 Testes de Precisão e Confiabilidade do Sistema

5.1.1 Teste de Precisão do ZMPT101B

Inicialmente, com o intuito de verificar a confiabilidade das medições do ZMPT101B, foi realizado a medição das tensões de saída do sensor de tensão por um osciloscópio digital. Para isso, o sensor ZMPT101B foi ligado à rede elétrica 127 V RMS e 60 Hz de frequência com alimentações de 3,3 V e 5,0 V.

Na Figura 5.1 pode ser observado as tensões de saída do sensor ZMPT101B para uma alimentação de 5 V. Na Figura 5.2 pode ser observado as tensões de saída do sensor ZMPT101B para uma alimentação de 3,3 V.

Como pode ser observado nas Figuras 5.1 e 5.2, as tensões são condizentes com as especificações do *datasheet*. No caso da alimentação de 5 V a tensão média de saída foi de 2,54 V, aproximadamente metade da tensão de alimentação. No caso da alimentação de 3,3 V a tensão média de saída foi de 1,69 V, aproximadamente metade da tensão de alimentação.

O valor de 0,8 V de pico a pico foi definido utilizando o potenciômetro de ajuste integrado junto a placa do ZMPT101B. Utilizando os valores mínimos e máximos é possível identificar que foi atribuído aproximadamente 0,4 V acima para os valores médios de tensão e 0,4 V abaixo dos valores médios de tensão para as análises de alimentação com 5,0 V e 3,3 V. Dessa forma, é possível ve-

rificar que o sistema se comporta da forma esperada para as medições da onda sinusoidal fornecida pela rede elétrica.

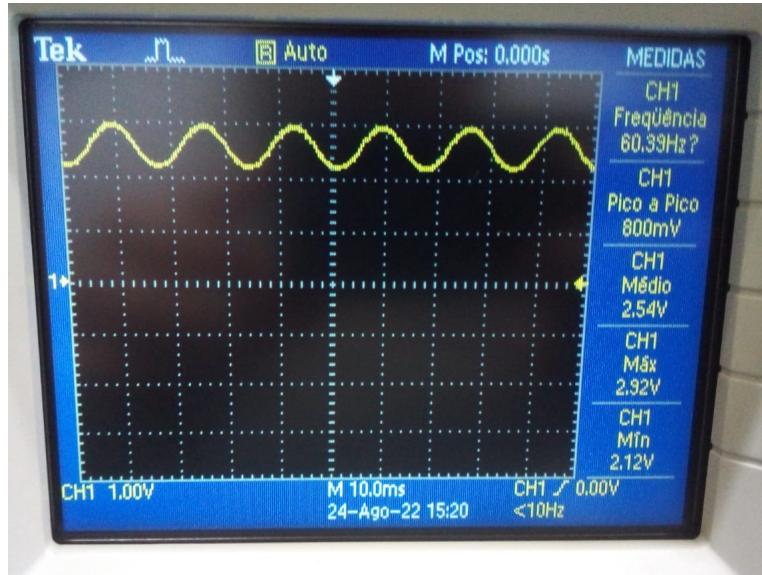


Figura 5.1: Teste de Confiabilidade do ZMPT101B com Alimentação de 5,0 V.

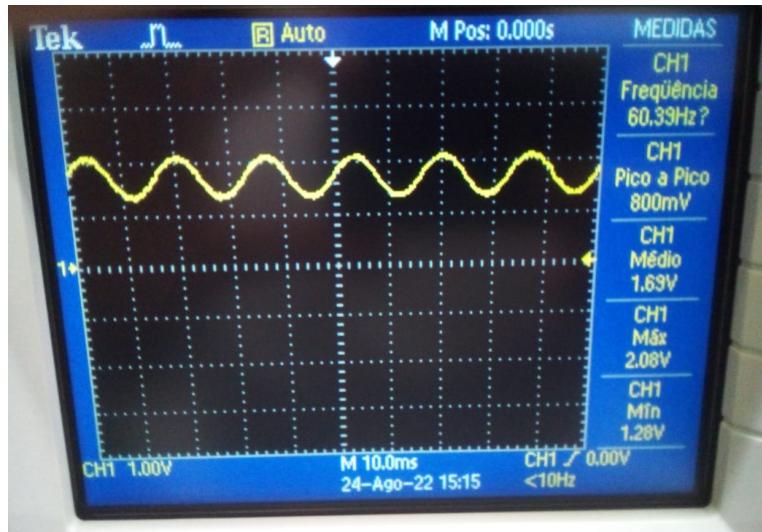


Figura 5.2: Teste de Confiabilidade do ZMPT101B com Alimentação de 3,3 V.

Para estabelecer os valores de referência da tensão de saída do ZMPT101B após a conversão analógico-digital do microcontrolador ESP32, explicitados na expressão (4.2), foi utilizado uma análise de leitura do sinal junto à porta série do microcontrolador. Como pode ser observado na Figura 5.3, o valor médio

de tensão da onda sinusoidal ficou em torno de 1870 e o valor máximo de 2200, sendo equivalente aos valores 0 V e 180 V, respectivamente.



Figura 5.3: Conversor analógico-digital do microcontrolador ESP32.

5.1.2 Teste de Precisão do ACS712T

Inicialmente, com o intuito de verificar a confiabilidade das medições do ACS712T, foi realizado a medição das tensões de saída do sensor de corrente por um osciloscópio digital. Para isso, o sensor ACS712T foi ligado à rede elétrica de 127 V RMS e 60 Hz de frequência em série com uma carga resistiva de 150 Ω e uma tensão de alimentação de 5,0 V. Na Figura 5.4 pode ser observado a imagem do osciloscópio utilizado no teste.

Como pode ser observado na Figura 5.4, o valor de tensão RMS é condizente com a tabela apresentada na folha de dados do fabricante, sendo a tensão RMS de saída metade do valor de VCC. Da mesma forma, o valor de tensão de pico a pico se mostrou concordante com o valor esperado, de acordo com o fabricante, em que 1 A representa uma alteração de 0,185 V na saída do sensor. Para o caso do teste realizado, a corrente do circuito era de 0,8467 A, algo que deveria representar uma tensão de pico a pico de 0,313 V na saída do sensor, um valor aproximado aos 0,320 V apresentados no osciloscópio.

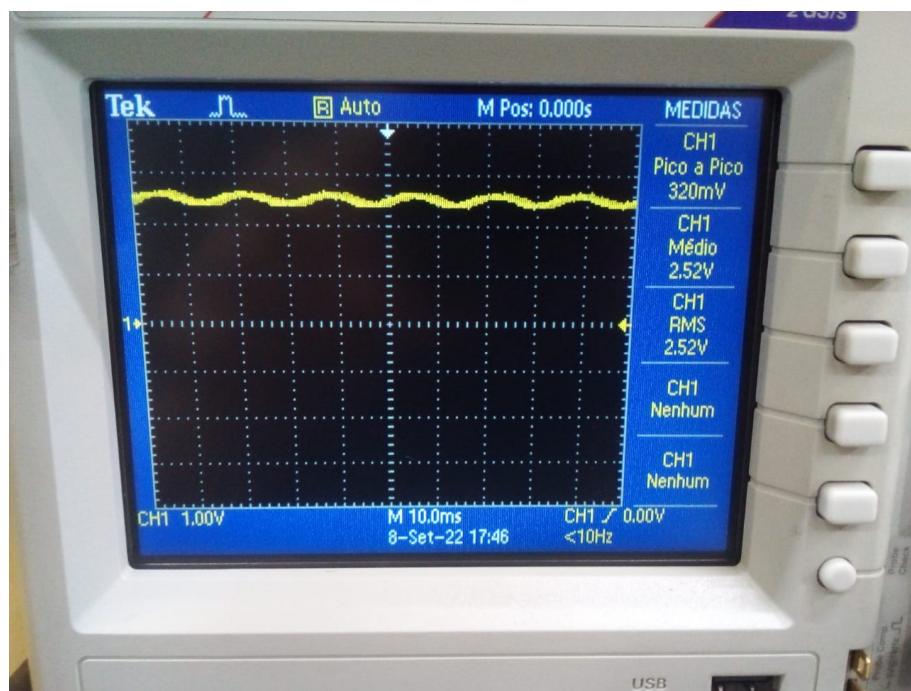


Figura 5.4: Teste de Confiabilidade do ACS712T.

5.2 Funcionamento Completo do Sistema *Smart Grid*

5.2.1 Funcionamento do Sistema *Smart Grid*

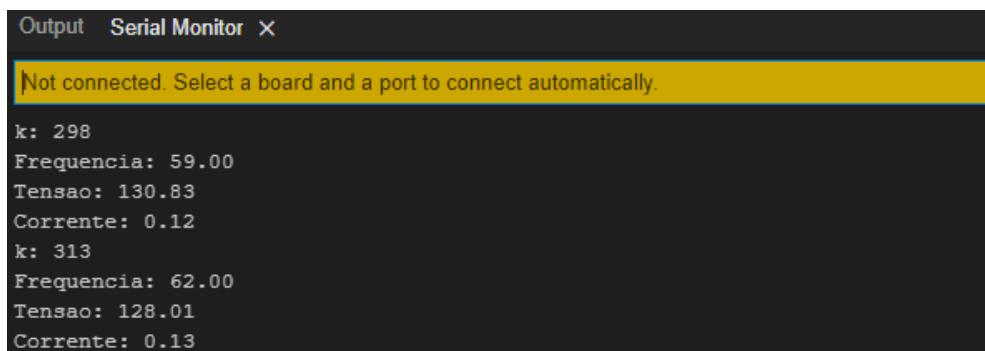
Para o funcionamento completo de uma rede *smart grid* com comunicação LoRa, foi realizado um longo processo que passa pela aquisição, serialização, tratamento, armazenamento e análise dos dados. O detalhamento desse processo pode ser dividido em algumas etapas:

- Aquisição dos dados por parte do *smart meter*;
- Transmissão dos dados do *smart meter* e recepção dos dados pelo *gateway* via rádio LoRa;
- Encaminhamento dos dados recebidos pelo *gateway* para um servidor MQTT via Wifi;
- Armazenamento dos dados recebidos pelo servidor MQTT em um banco SQL;
- Monitoramento dos dados armazenados no banco SQL através de uma visualização gráfica;
- Análise estatística dos dados através de algoritmos de inteligência artificial.

Cada uma dessas etapas do processo realizadas no protótipo de *smart grid* serão detalhadas a seguir.

5.2.2 Aquisição dos Dados por parte do *Smart Meter*

Para a aquisição dos dados, como dito anteriormente, foi construído um protótipo de *smart meter* que possui sensores de tensão e corrente, em que os dados desses sensores são convertidos em valores digitais pelo microprocessador do sistema através de um ADC. A Figura 5.5 mostra um exemplo de uma leitura de dados do *smart meter* através da porta série do microcontrolador ESP32.



```

Output  Serial Monitor X

Not connected. Select a board and a port to connect automatically.

k: 298
Frequencia: 59.00
Tensao: 130.83
Corrente: 0.12
k: 313
Frequencia: 62.00
Tensao: 128.01
Corrente: 0.13
  
```

Figura 5.5: Log dos Dados através do Monitor Série.

5.2.3 Transmissão dos Dados do *Smart Meter* e Recepção dos Dados pelo *Gateway* via Rádio LoRa

Como dito anteriormente, foi utilizado um rádio LoRa para a transmissão e recepção dos dados entre os protótipos *smart meter* e *gateway*.

O trecho de código ilustrado na Figura 5.6 contido no *software* do *smart meter* é responsável pela transmissão dos dados para o *gateway*.

```

// Lora
Serial2.println(tensao_rms);
Serial2.println(corrente_rms);
Serial2.println(freq);
  
```

Figura 5.6: Código Responsável pela Transmissão dos Dados.

O trecho de código ilustrado na Figura 5.7 contido no *software* do *gateway* é responsável pela recepção dos dados enviados pelo *smart meter*.

```

if (Serial2.available()){
    recebido = Serial2.readString();
    Serial.println(recebido);
    digitalWrite(ledPin, HIGH);
}

```

Figura 5.7: Código Responsável pela Recepção dos Dados.

5.2.4 Encaminhamento dos Dados Recebidos pelo *Gateway* para um Servidor MQTT via Wifi

Para o encaminhamento dos dados recebidos pelo *gateway* para um servidor MQTT, foi necessário o estabelecimento de uma conexão Wifi por parte do *gateway*. A Figura 5.8 ilustra o exemplo da conexão Wifi realizada pelo *gateway*.

```

Connecting to WiFi..
Connecting to WiFi..
Connected to the WiFi network
MacAddress: 24:0A:C4:8B:5D:50
Endereço de IP: 192.168.0.17

```

Figura 5.8: Exemplo de Conexão Wifi do *Gateway*.

Após a conexão Wifi foi necessário o estabelecimento de um servidor *web* para disponibilização dos dados via HTTP. O trecho do código do *gateway* responsável por esse processo pode ser observado na Figura 5.9.

```

WiFiClient client = server.available();
if (client) {
    Serial.println("New Client.");
    while (client.connected()) {
        if (client.available()) {
            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html");
            client.println();
            if (recebido != "") {
                client.print(recebido);
            }
            break;
        }
    }
}

```

Figura 5.9: Código Responsável pelo Estabelecimento do Servidor *web*.

Com o servidor *web* estabelecido, o código Python contido no Apêndice C é responsável por realizar a publicação desses dados no servidor MQTT. Na Figura 5.10 pode ser observado um exemplo de *log* do código Python realizando as publicações no servidor MQTT.

```
C:\Users\Vinicius\Desktop\ScriptsPython>python pub_mqtt.py
Aguardando conexão com o servidor.
['248.23', '0.15', '42.00']
Dados Publicados

C:\Users\Vinicius\Desktop\ScriptsPython>python pub_mqtt.py
Aguardando conexão com o servidor.
['250.35', '0.01', '46.00']
Dados Publicados

C:\Users\Vinicius\Desktop\ScriptsPython>python pub_mqtt.py
Aguardando conexão com o servidor.
['255.30', '0.05', '54.00']
Dados Publicados

C:\Users\Vinicius\Desktop\ScriptsPython>python pub_mqtt.py
Aguardando conexão com o servidor.
['255.30', '0.05', '54.00']
Dados Publicados
```

Figura 5.10: *Log* de Mensagens na Publicação MQTT.

5.2.5 Armazenamento dos Dados Recebidos pelo Servidor MQTT em um Banco SQL

Para o armazenamento dos dados publicados no servidor MQTT em um banco de dados SQL, como dito anteriormente, foi utilizado um *script* em Python. Este é responsável por realizar a subscrição no tópico desejado no servidor MQTT e realizar a inserção das mensagens no banco de dados sempre que ocorra uma publicação. O *script* pode ser analisado por completo no Apêndice D.

As Figuras 5.11 e 5.12 apresentam os exemplos de *log* de subscrição em todos os tópicos do servidor MQTT via Mosquitto e *log* do *script* de armazenamento.

```
C:\Program Files\mosquitto>mosquitto_sub -p 1884 -t #
127.30 0.13 59.00 2022-08-09 16:49:53.967405
126.40 0.13 58.00 2022-08-09 16:54:15.691516
126.70 0.13 59.00 2022-08-09 17:00:51.045712
127.30 0.13 60.00 2022-08-09 17:01:27.458185
```

Figura 5.11: *Log* de Subscrição em Todos os Tópicos via Mosquitto.

```
C:\Users\Vinicius\Desktop\ScriptsPython>python armazenaBD.py
creating new instance
connecting to broker
Subscribing to topic smartmeter1
message received 126.40 0.13 58.00 2022-08-09 16:54:15.691516
message topic= smartmeter1
message qos= 0
message retain flag= 0
message received 126.70 0.13 59.00 2022-08-09 17:00:51.045712
message topic= smartmeter1
message qos= 0
message retain flag= 0
```

Figura 5.12: Log do *Script* de Armazenamento de Dados no Banco SQL.

5.2.6 Monitoramento dos Dados Armazenados no Banco SQL através de uma Visualização Gráfica

Para realizar o monitoramento e a visualização gráfica dos dados armazenados no Banco SQL, foi programado um *script* em Python responsável por realizar a visualização dos dados. O código completo pode ser analisado no Apêndice E.

As Figuras 5.13 e 5.14 mostram o resultado de alguns exemplos de visualizações realizadas pelo *script* Python, ilustrando gráficos de tensão por horário e corrente por horário, respectivamente.

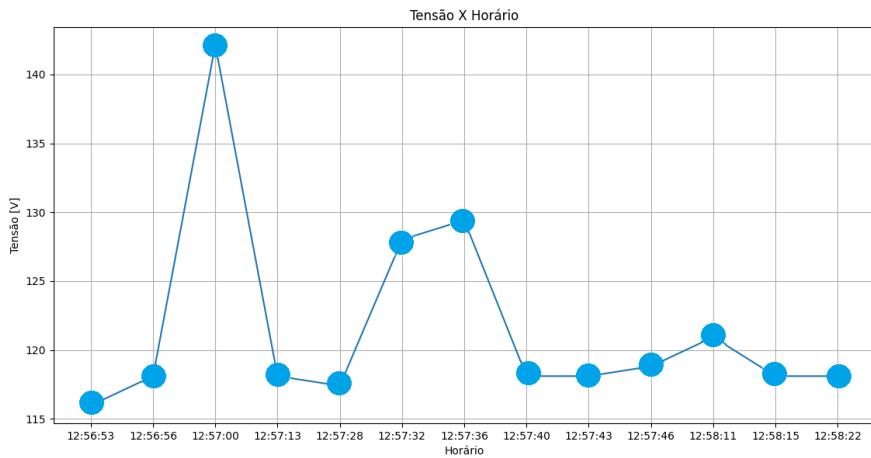


Figura 5.13: Gráfico de Tensão por Horário.

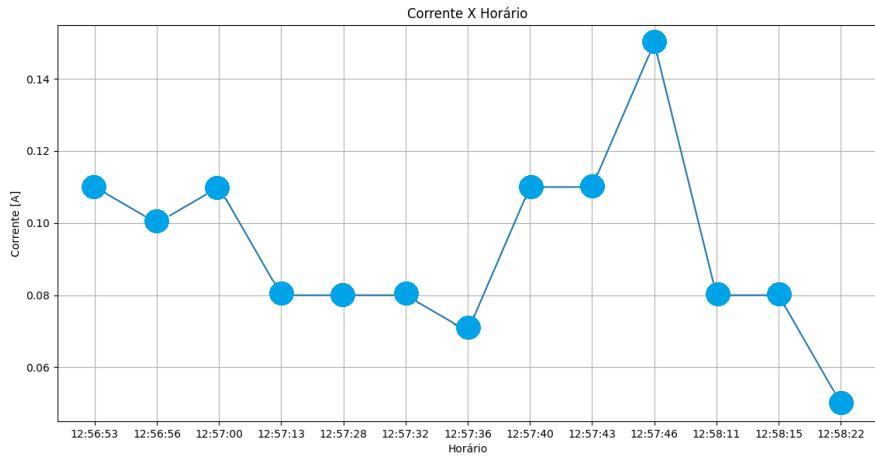


Figura 5.14: Gráfico de Corrente por Horário.

5.2.7 Análise Estatística dos Dados através de Algoritmos de Inteligência Artificial

Assim como comentado anteriormente, devido ao protótipo de rede *smart grid* construído não possuir dados de coletas de valores reais, o banco usado para a análise estatística dos dados de consumo e uso de algoritmos de inteligência artificial para previsão de demanda foi populado usando dados reais do Ministério de Minas e Energias (MME) [50].

Na Figura 5.15 pode ser observado um exemplo do gráfico gerado pelo algoritmo de previsão de carga. Este representa a curva de carga residencial do estado de São Paulo junto da previsão do algoritmo para o ano de 2021.

Na Figura 5.16 pode ser observado o MAPE entre os valores previstos e os valores reais.

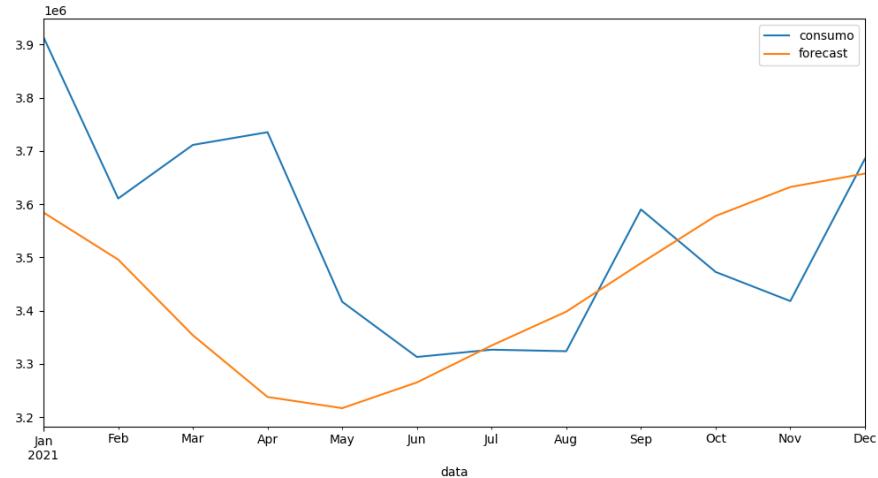


Figura 5.15: Gráfico Curva de Carga Versus Previsão de Carga.

	consumo	forecast	MAPE
data			
2021-01-01	3914071.87	3.584460e+06	8.421207
2021-02-01	3610851.55	3.496101e+06	3.177944
2021-03-01	3711561.88	3.353899e+06	9.636441
2021-04-01	3735481.77	3.237927e+06	13.319690
2021-05-01	3416755.88	3.216904e+06	5.849182
2021-06-01	3313095.80	3.265303e+06	1.442532
2021-07-01	3326780.92	3.334595e+06	-0.234895
2021-08-01	3323903.41	3.398275e+06	-2.237490
2021-09-01	3590325.44	3.489411e+06	2.810723
2021-10-01	3472833.08	3.578056e+06	-3.029878
2021-11-01	3418145.80	3.632447e+06	-6.269527
2021-12-01	3685680.57	3.657745e+06	0.757957

Figura 5.16: Análise Estatística da Previsão de Carga Consumidora.

Capítulo 6

Conclusões

6.1 Considerações Finais

6.1.1 Objetivos Alcançados

No corrente trabalho foi apresentado um estudo sobre as redes elétricas inteligentes: suas principais vantagens em relação a uma rede elétrica tradicional e suas dificuldades para implementação. Além disso, foi detalhado um modelo protótipo de REI utilizando como forma de comunicação uma rede LPWAN. O modelo protótipo cumpriu todo o trajeto dos dados de uma REI ideal, partindo na coleta dos parâmetros da rede até a visualização e análise estatística dos dados pelas concessionárias.

Para a análise dos dados, foi utilizado um algoritmo de inteligência artificial *Long Short Term Memory* (LSTM) para previsão de séries temporais. De acordo com o referencial teórico o algoritmo LSTM apresentou ser adequado para a análise do tipo de dados coletados pelo *smart meter*.

6.1.2 Dificuldades

As principais dificuldades encontradas no projeto encontram-se na prototipação do *smart meter*, mais especificamente na escolha dos sensores e seus ajustes de precisão. Como comentado no capítulo de *hardware*, a precisão dos sensores foi ajustada realizando testes práticos após a construção do protótipo *smart meter*. Essa dificuldade ocorre devido a característica de o sinal de saída dos sensores de tensão e de corrente oferecerem como *offset* a metade da tensão de alimentação dos mesmos, tensão esta que foi fornecida pelo microcontrolador e pode sofrer alteração de acordo com a quantidade de componentes alimentados.

Além disso, como comentado no capítulo de *software*, a definição da frequência da rede foi estabelecida pelo *smart meter* utilizando um algoritmo de contagem de vezes que a sinusóide da tensão atingia o valor de 0 V. Dessa forma, o valor após a conversão analógica-digital realizada pelo microcontrolador referente a 0 V da rede teve de ser obtido experimentalmente, assim como o intervalo de erro para uma amostra ser considerada como 0 V.

6.1.3 Aplicações Práticas e Trabalhos Futuros

Para aplicações reais do modelo de *smart grid* com comunicação LoRa apresentado, sugere-se uma análise dos sensores de tensões e corrente mais detalhada, especificamente se tratando do sensor de corrente, aconselha-se o uso de um sensor invasivo devido a capacidade de operar em correntes mais elevadas e também a maior robustez física do componente.

Além disso, para uma aplicação prática do modelo sugere-se a utilização de uma fonte reguladora de tensão interna ao *smart meter*, podendo ser utilizado um transformador convencional ou de preferência uma fonte chaveada devido a ser mais leve e compacta. Em ambos os casos, mas principalmente se a escolha seja uma fonte chaveada, é necessário atentar-se com a geração de distorções harmônicas na rede elétrica, efeito esse que pode interferir na qualidade da rede, e consequentemente na precisão das medições [51].

Dessa forma, tratando-se das distorções harmônicas para uma aplicação profissional do modelo, torna-se adequado a utilização de filtros entre a rede elétrica e os sensores de tensão e corrente, com o objetivo de minimizar os erros de medição [51].

Por fim, para uma aplicação prática do modelo com comunicação LoRa e protocolo LoRaWAN, torna-se necessário a pesquisa dos espectros de frequências de comunicação permitidos para a comunicação LoRa em cada região do mundo.

Referências

- [1] Henrique Mattede, “Topologia de um sep.” Disponível em: <https://www.mundodaeletrica.com.br/um-pouco-mais-sobre-o-sistema-eletrico-de-potencia-sep/>. Accessed: 2022-07-07. [Quoted on p. ix, 10]
- [2] David Mayne, “Topologia de uma rei.” Disponível em: <https://br.mouser.com/applications/smart-grid-energize-world/>. Accessed: 2022-07-07. [Quoted on p. ix, 11]
- [3] R. M. d. S. B. J. S. B. M. Flávio Galvão Calhau, Flávia Maristela Santos Nascimento, “Smart grids: Características, requisitos e perspectivas.” https://www.researchgate.net/publication/267568184_Smart_Grids_Caracteristicas_Requisitos_e_Perspectivas. Accessed: 2022-07-07. [Quoted on p. ix, 9, 13, 14]
- [4] A. M. D. S. ALONSO, “Smart grids: tecnologias de comunicação e sua realidade no brasil.” <http://www.em.ufop.br/images/MonografiasControleAutomacao/2014/AugustoMatheusDosSantosAlonso.pdf>. Accessed: 2022-06-30. [Quoted on p. ix, 5, 15]
- [5] D. Patel, “Low power wide area networks (lpwan): Technology review and experimental study on mobility effect.” <https://openprairie.sdsstate.edu/etd/2667/>. Accessed: 2022-07-26. [Quoted on p. ix, 16, 17, 18, 19]
- [6] J. B. J. F. Phui San Cheong, Chris Hawinkel, “Comparison of lorawan classes and their power consumption.” https://www.researchgate.net/publication/321698801_Comparison_of_LoRaWAN_Classes_and_their_Power_Consumption. Accessed: 2022-07-28. [Quoted on p. ix, 21, 22]
- [7] “Cap sistemas.” <https://capsistema.com.br/index.php/2020/11/07/tutorial-do-protocolo-mqtt-descricao-tecnica-com-exemplo-pratico-de-mosquitto/>. Accessed: 2022-08-08. [Quoted on p. ix, 23]

- [8] E. P. Energética, "Painel de monitoramento do consumo de energia elétrica." <https://www.epe.gov.br/pt/publicacoes-dados-abertos/publicacoes/consumo-de-energia-eletrica>. Accessed: 2022-10-18. [Quoted on p. ix, 24, 25]
- [9] C. H. V. VIVIAN, "Estudo de caso da curva de consumidor comercial." <https://www.lume.ufrgs.br/bitstream/handle/10183/134912/000988036.pdf?sequence=1>. Accessed: 2022-10-16. [Quoted on p. ix, 25]
- [10] M. Taştan, "Internet of things based smart energy management for smart home," *KSII Transactions on Internet and Information Systems*, vol. 13, p. 18, 06 2019. Accessed: 2022-08-18. [Quoted on p. ix, 29]
- [11] "Datasheet sensor de corrente acs712t." <https://5nr0rwxhmqqijik.leadongcdn.com/ZMPT101B+specification-aidijBqoKomRilSqqokpjkp.pdf>. Accessed: 2022-08-18. [Quoted on p. ix, 31]
- [12] "Pinout sensor de corrente acs712t." <https://components101.com/sensors/acs712-current-sensor-module>. Accessed: 2022-08-18. [Quoted on p. ix, 31]
- [13] "Pinout microcontrolador esp32 devkit v1." <https://www.mischianti.org/2021/02/17/doit-esp32-dev-kit-v1-high-resolution-pinout-and-specs/>. Accessed: 2022-08-18. [Quoted on p. ix, 32]
- [14] "Pinout da placa lora e32-433t20d." <https://www.mischianti.org/2019/10/15/lora-e32-device-for-arduino-esp32-or-esp8266-specs-and-basic-usage-part-1/>. Accessed: 2022-08-23. [Quoted on p. ix, 33]
- [15] R. Dathein, "Inovação e revoluções industriais: uma apresentação das mudanças tecnológicas determinantes nos séculos xviii e xix." <http://www.ufrgs.br/ufrgs/inicial>. Accessed: 2022-05-17. [Quoted on p. 2]
- [16] "A brief history of home electrical wiring." <https://www.thespruce.com/how-old-is-your-wiring-1152880>. Accessed: 2022-05-18. [Quoted on p. 2]
- [17] L. d. C. Rosiel Camargo Souza, Rodrigo Diogo Dahmer, "Redes elétricas inteligentes: Evolução das redes convencionais." <http://www.api.org.br/conferences/index.php/ENPI2017/ENPI2017/paper/viewFile/152/107>. Accessed: 2022-05-18. [Quoted on p. 2]
- [18] S. D. Fugita, "Smart meter integrado a analisador de qualidade de energia para propósitos de identificação de cargas residenciais." <http://>

- br/jspui/simple-search?query=paulo+barnabe. Accessed: 2022-07-07. [Quoted on p. 11]
- [28] D. B. Marques, "Metodologia para análise da dependabilidade de smart grids." https://repositorio.ufrn.br/bitstream/123456789/20927/1/DanielleBritoMarques_DISSSERT.pdf. Accessed: 2022-07-07. [Quoted on p. 11]
- [29] I. T. Ricardo Rivera, Alexandre Siciliano Esposito, "Redes elétricas inteligentes (smart grid): oportunidade para adensamento produtivo e tecnológico local." <http://www.provedor.nuca.ie.ufrj.br/eletrobras/estudos/rivera1.pdf>. Accessed: 2022-07-07. [Quoted on p. 12]
- [30] Z. L. Fang Yang, Xianyong Feng, "Advanced microgrid energy management system for future sustainable and resilient power grid." file:///C:/Users/Vinicius/Downloads/fy_microgrid_xiangyong_final1-referencefile.pdf. Accessed: 2022-07-08. [Quoted on p. 12]
- [31] A. F. d. S. C. João Victor de Andrade Leite, "Estudo e aplicação da smart grid no sistema elétrico de distribuição brasileiro." <https://revistas.unifacs.br/index.php/index/search/advancedResults>. Accessed: 2022-07-08. [Quoted on p. 13]
- [32] H. Q. P. J. Maria Carolina Avelar Fadul Ferreira, "Perspectivas e desafios para a implantação das smart grids: um estudo de caso dos eua, portugal e brasil." <https://pantheon.ufrj.br/bitstream/11422/2457/1/MCAFFerreira.pdf>. Accessed: 2022-07-08. [Quoted on p. 13, 14]
- [33] A. A. S. Cássia C. Silva Pereira, Ruy de Oliveira, "Oportunidades e desafios smart grid." <http://jornada.cba.ifmt.edu.br/jornada/index.php/jornada2013/jornada2013/paper/viewFile/77/44>. Accessed: 2022-07-12. [Quoted on p. 13]
- [34] A. V. A. O. Yeliz Yoldaş, S M Muyeen, "Enhancing smart grid with microgrids: Challenges and opportunities." https://www.researchgate.net/publication/312367566_Enhancing_smart_grid_with_microgrids_Challenges_and_opportunities. Accessed: 2022-07-08. [Quoted on p. 14]
- [35] M. J. L. V. R. da Silva, "Smart grids em portugal plano de negócio para serviço de planeamento e gestão remota de consumos eléctricos." <https://fenix.tecnico.ulisboa.pt/cursos/meec/dissertacao/2353642347241>. Accessed: 2022-07-08. [Quoted on p. 14]
- [36] A. Z. M. Z. Marco Centenaro, Lorenzo Vangelista, "Long-range communications in unlicensed bands: the rising stars in the iot and smart city scenarios." https://www.researchgate.net/publication/282603371_

- Long-Range_Communications_in_Unlicensed_Bands_the_Rising_Stars_in_the_IoT_and_Smart_City_Scenarios. Accessed: 2022-07-26. [Quoted on p. 19]
- [37] M. S. Usman Raza, Parag Kulkarni, "Low power wide area networks: An overview." <https://arxiv.org/pdf/1606.07360.pdf>. Accessed: 2022-07-26. [Quoted on p. 20]
- [38] F. Sforza, "United states patent: Us 8.406.275 b2." <https://patents.google.com/patent/US8406275>. Accessed: 2022-07-28. [Quoted on p. 20]
- [39] "Semtech corporation." <https://www.semtech.com/>. Accessed: 2022-07-28. [Quoted on p. 20]
- [40] G. F. Nikoleta Andreadou, Miguel Olariaga Guardiol, "Telecommunication technologies for smart grid projects with focus on smart metering applications." <https://www.mdpi.com/1996-1073/9/5/375>. Accessed: 2022-07-28. [Quoted on p. 21]
- [41] G. Santos, "Protocolo mqtt: O que é, como funciona e vantagens." <https://www.automacaoindustrial.info/author/admin/>. Accessed: 2022-08-08. [Quoted on p. 22, 23]
- [42] R. J. C. I. V. d. C. M. Fábio Cássio de Sousa, Ricardo Donizeti Soares, "Iiot utilizando protocolo mqtt." <https://repositorio.animaeducacao.com.br/simple-search?query=Fabio+Cassio+de+Sousa>. Accessed: 2022-08-08. [Quoted on p. 22]
- [43] "Site oficial eclipse mosquitto." <https://mosquitto.org/>. Accessed: 2022-08-08. [Quoted on p. 24]
- [44] L. A. A. de Oliveira, "Tratamento de dados de curvas de carga via anÁlise de agrupamentos e transformada wavelets." <https://www.cos.ufrj.br/uploadfile/1389010456.pdf>. Accessed: 2022-10-14. [Quoted on p. 24, 25]
- [45] H. Madsen, "Time series analysis." file:///C:/Users/Vinicius/Downloads/time_series_introduction.pdf. Accessed: 2022-10-16. [Quoted on p. 26]
- [46] P. M. Alves, "Inteligência artificial e redes neurais." <https://www.ipea.gov.br/cts/pt/central-de-conteudo/artigos/artigos/106-inteligencia-artificial-e-redes-neurais>. Accessed: 2022-10-18. [Quoted on p. 26]
- [47] W. P. A. Jhonatan Correa Leandro, "Aplicação de redes neurais lstm para previsão de séries temporais financeiras." <https://repositorio.ufgd.edu.br/handle/10662/10000>.

- edu.br/jspui/bitstream/prefix/4806/1/JhonatanCorreaLeandro.pdf. Accessed: 2022-10-18. [Quoted on p. 26]
- [48] Q. Z. L. Electronic, “*Datasheet* sensor de tensão zmpt101b.” <https://www.sparkfun.com/datasheets/BreakoutBoards/0712.pdf>. Accessed: 2022-08-18. [Quoted on p. 29]
- [49] “*Datasheet* microcontrolador esp32 espressif.” https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf. Accessed: 2022-08-19. [Quoted on p. 32]
- [50] M. de Minas e Energia, “Consumo de energia elétrica no brasil.” https://basedosdados.org/dataset/br-mme-consumo-energia-eletrica?bdm_table=uf. Accessed: 2022-09-27. [Quoted on p. 59]
- [51] P. Deckmann, “Avaliação da qualidade da energia elétrica.” <https://www.dsce.fee.unicamp.br/~antenor/pdffiles/qualidade/a5.pdf>. Accessed: 2022-09-28. [Quoted on p. 62]

Apêndice A - Código *Smart Meter*

```
#include <LiquidCrystal_I2C.h>

// Portas Seriais LoRa
#define RXD2 16
#define TxD2 17
bool estado = 0;

// Display Lcd
LiquidCrystal_I2C lcd(0x27, 16, 2);

// Fun o Millis
unsigned long startTime;
unsigned long currentTime;
const unsigned long period = 5000;

// Variaveis Sensor de Tens o
int pin_tensao = 36;
int maior_valor = 0;
int tensao_inst;
float Vpp;
float tensao_rms;
int i = 0;

// Variaveis Sensor de Corrente
int pin_corrente = 34;
int maior_valor_cor = 0;
int corrente_inst;
float Ipp;
float corrente_rms;
int j = 0;
```

```
// Variaveis Frequencia da Rede
double freq = 0;
int k = 0;

void setup() {
    pinMode(pin_tensao, INPUT);
    Serial.begin(9600);
    Serial2.begin(9600, SERIAL_8N1, RXD2, TXD2);
    lcd.init();
    lcd.setBacklight(HIGH);
    delay(500);
    startTime = millis();
}

void loop() {
    // -----Start Millis() -----
    currentTime = millis();

    // ----- Sensor de Tensao -----
    // Definicao do maior valor
    tensao_inst = analogRead(pin_tensao);
    if(tensao_inst > 1868 && tensao_inst < 1872){
        k = k + 1;
        //delay(1);
    }
    if (i < 30000){
        if (maior_valor <= tensao_inst){
            maior_valor = tensao_inst;
        }
    }else{
        Vpp = map(maior_valor,1870,2200,0,180);
        tensao_rms = Vpp/1.414;
        //Serial.print("Tens o de pico: ");
        //Serial.println(Vpp);
        //Serial.print("Tens o RMS: ");
        //Serial.println(tensao_rms);

        // Display no LCD
        lcd.setCursor(9,0);
        //lcd.print("Vp ");
        //lcd.setCursor(12,0);
        lcd.print(Vpp);
        lcd.setCursor(9,1);
        //lcd.print("Vr ");
        //lcd.setCursor(12,1);
        lcd.print(tensao_rms);
```

```
//Serial.println(maior_valor);
maior_valor = 0;
i = 0;
}

// Contador Sensor Tens o
i++;

// ----- Sensor de Corrente -----
// Definicao do maior valor
corrente_inst = analogRead(pin_corrente);
if (j < 30000){
    if (maior_valor_cor <= corrente_inst){
        maior_valor_cor = corrente_inst;
    }
} else{
    Ipp = map(maior_valor_cor ,1938 ,2655 ,0 ,5);
    //Ipp = ((maior_valor_cor - 1938) * 0.2) / 15;
    corrente_rms = Ipp/1.414;
    //Serial.print("Corrente de pico: ");
    //Serial.println(Ipp);
    //Serial.print("Corrente RMS: ");
    //Serial.println(corrente_rms);

    // Display no LCD
    lcd.setCursor(0,0);
    //lcd.print("Ip ");
    //lcd.setCursor(3,0);
    lcd.print(Ipp);
    lcd.setCursor(0,1);
    //lcd.print("Ir ");
    //lcd.setCursor(3,1);
    lcd.print(corrente_rms);

    //Serial.println(maior_valor_cor);
    maior_valor_cor = 0;
    j = 0;
}

// Contador Sensor Corrente
j++;

currentTime = millis();
if (currentTime - startTime >= period)
{
    freq = (k*1000) / 5000;
```

```
Serial.print("k: ");
Serial.println(k);
Serial.print("Frequencia: ");
Serial.println(freq);
Serial.print("Tensao: ");
Serial.println(tensao_rms);
Serial.print("Corrente: ");
Serial.println(corrente_rms);
k = 0;

// Lora
Serial2.println(tensao_rms);
Serial2.println(corrente_rms);
Serial2.println(freq);

startTime = currentTime;
}

}
```

Apêndice B - Código *Gateway*

```
#include <WiFi.h>
#include <HTTPClient.h>

WiFiServer server(80);
String recebido;

#include <HTTPClient.h>

// Novas portas Serials
#define RXD2 16
#define TxD2 17

// Funcao Millis
unsigned long startTime;
unsigned long currentTime;
const unsigned long period = 5000;

// Usuario da rede Wifi
const char* ssid = "*****";
// Senha da rede Wifi
const char* password = "*****";

void setup() {

    Serial.begin(9600);
    Serial2.begin(9600, SERIAL_8N1, RxD2, TxD2);

    startTime = millis();

    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.println("Connecting to WiFi..");
    }
}
```

```
}

Serial.println("Connected to the WiFi network");
WiFi.mode(WIFI_MODE_STA);
Serial.print("MacAddress: ");
Serial.println(WiFi.macAddress());
Serial.print("Endereco de IP: ");
Serial.println(WiFi.localIP());
server.begin();

}

void loop() {

    // ----- Start LoRa -----
    if (Serial2.available()){
        recebido = Serial2.readString();
        Serial.println(recebido);
    }
    // ----- End LoRa -----


    WiFiClient client = server.available();
    if (client) {
        Serial.println("New Client.");
        while (client.connected()) {
            if (client.available()) {
                client.println("HTTP/1.1 200 OK");
                client.println("Content-type:text/html");
                client.println();
                if (recebido != ""){
                    client.print(recebido);
                }
                break;
            }
        }
    }

    // ----- Start Millis() -----
    currentTime = millis();
    if (currentTime - startTime >= period)
    {
        startTime = currentTime;
    }
    // ----- End Millis() -----
}
```

Apêndice C - Código MQTT

```
import requests as req
import paho.mqtt.client as paho
import datetime

# Url do Servidor ESP32 com os dados dos sensores
url = 'http://172.16.38.100'

# Define cabe alho HTTP
headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0",
    "Accept-Encoding": "*",
    "Connection": "keep-alive"
}

print("Aguardando conexão com o servidor.")

x = 0
while(x == 0):
    try:
        # Recebe o body da pagina web
        response = req.get(url, headers=headers)
        # Divide o body do html em uma string por espaçoamento.
        parametro = response.text.split()
        print(response.text.split())
        x = 1
    except:
        x = 0

# Adiciona data e hora no pacote
horario = str(datetime.datetime.now())
parametro.append(horario)

# Define parametros para a conexão com o broker MQTT
```

```
broker="localhost"
port=1884

# Funcao para callback
def on_publish(client,userdata,result):
    print("Dados Publicados")
    pass

# Define objeto cliente
client1 = paho.Client("medidor1")

# Define fun  o para callback
client1.on_publish = on_publish

# Inicia conexao com o broker
client1.connect(broker,port)

# Realiza a publica  o no t pico "smartmeter1".
client1.publish(
    "smartmeter1",
    parametro[0] + " " +
    parametro[1] + " " +
    parametro[2] + " " +
    parametro[3]
)
```

Apêndice D - Código Banco de Dados

```
import paho.mqtt.client as paho
import sqlite3

con = sqlite3.connect('bancoDeDados.db')
cur = con.cursor()

def salvaBD(mensagem):
    m = mensagem.split()
    lista = [
        (m[3] + " " + m[4], m[0], m[1], m[2]),
    ]
    cur.executemany("insert into registros values (?, ?, ?, ?)", lista)
    con.commit()

def on_message(client, userdata, message):
    print("message received ", str(message.payload.decode("utf-8")))
    print("message topic=", message.topic)
    print("message qos=", message.qos)
    print("message retain flag=", message.retain)
    salvaBD(str(message.payload.decode("utf-8")))

broker_address="localhost"
print("creating new instance")
client = paho.Client("medidor")
print("connecting to broker")
client.connect(broker_address, 1884)
print("Subscribing to topic","smartmeter1")
client.on_message = on_message
client.subscribe("smartmeter1",1)

# Manter o script sempre ligado para caso de publicacoes
```

```
client.loop_forever()  
con.close()
```

Apêndice E - Código *Dashboard*

```
import numpy as np
import sqlite3
import matplotlib.pyplot as plt

con = sqlite3.connect('bancoDeDados.db')
cur = con.cursor()

sqlite_select_query = """SELECT * from registros"""
cur.execute(sqlite_select_query)
records = cur.fetchall()

cur.close()

con.close()

data = []
tensao = []
corrente = []
frequencia = []

for row in records:
    print("Data: ", row[0])
    print("Tensão: ", row[1])
    print("Corrente: ", row[2])
    print("Frequência: ", row[3])
    print("\n")
    data.append(row[0])
    tensao.append(row[1])
    corrente.append(row[2])
    frequencia.append(row[3])

def listaParaDecimal(lista):
    listaDecimal = []
```

```
for item in lista:
    listaDecimal.append(float(item))
return listaDecimal

def separaData(datas):
    listaFinalDatas = []
    for d in datas:
        listaDatas = d.split()
        finalDatas = listaDatas[1]
        listaFinalDatas.append(finalDatas[:-7])
    return listaFinalDatas

tensaoF = listaParaDecimal(tensao)
correnteF = listaParaDecimal(corrente)
frequenciaF = listaParaDecimal(frequencia)
datasF = separaData(data)

plt.figure(1)
plt.title("Corrente X Hor rio")
plt.xlabel("Hor rio")
plt.ylabel("Corrente [A]")
plt.grid()
plt.plot(datasF, correnteF)
plt.show()

plt.figure(2)
plt.title("Tens o X Hor rio")
plt.xlabel("Hor rio")
plt.ylabel("Tens o [V]")
plt.grid()
plt.plot(datasF, tensaoF)
plt.show()
```

Apêndice F - Código de Padronização do Banco de Dados

```
import pandas as pd
import matplotlib.pyplot as plt

# Site para aquisicao do banco de dados:
# https://basedosdados.org/dataset/br-mme-consumo
# -energia-eletrica?bdm_table=uf

df = pd.read_csv('uf.csv')

# Definicao do tipo de consumo a ser analisado
def defineTipoConsumo():
    print("Escolha o tipo de consumo: ")
    print("1 - Residencial")
    print("2 - Comercial")
    print("3 - Industrial")
    print("4 - Total")
    tipo_consumoF = int(input("Digite o numero: "))
    return tipo_consumoF

# Definicao do tipo de consumo a ser analisado
validacao = False
while validacao == False:

    try:
        tipo_consumo = defineTipoConsumo()
    except:
        print("Valor deve ser um numero inteiro!")

    if tipo_consumo == 1:
        tipoC = "Residencial"
```

```
df = df[df.tipo_consumo == tipoC]
validacao = True
elif tipo_consumo == 2:
    tipoC = "Comercial"
    df = df[df.tipo_consumo == tipoC]
    validacao = True
elif tipo_consumo == 3:
    tipoC = 'Industrial'
    df = df[df.tipo_consumo == tipoC]
    validacao = True
elif tipo_consumo == 4:
    tipoC = 'Total'
    df = df[df.tipo_consumo == tipoC]
    validacao = True
else:
    print("Valor invalido!")

# Define estado como SP
df = df[df.sigla_uf == 'SP']

# Formata a data
data = pd.date_range(start='01/15/2004', end='12/15/2021', periods=len(df))
df['data'] = data.strftime('%Y-%m')
df = df.set_index('data')
df.index.freq='MS'

# Deleta coluna de numero de consumidores
df.drop(["numero_consumidores"], axis=1, inplace=True)
# Deleta coluna ano
df.drop(["ano"], axis=1, inplace=True)
# Deleta coluna mes
df.drop(["mes"], axis=1, inplace=True)
# Deleta coluna tipo_consumo
df.drop(["tipo_consumo"], axis=1, inplace=True)
# Deleta coluna sigla_uf
df.drop(["sigla_uf"], axis=1, inplace=True)
# Converte a tabela para .CSV
df.to_csv('Ai.csv')
```

Apêndice G - Código Previsão de Carga

```
from cmath import sqrt
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from statsmodels.tsa.seasonal import seasonal_decompose
from sklearn.preprocessing import MinMaxScaler
from keras.preprocessing.sequence import TimeseriesGenerator
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM

df = pd.read_csv('Ai.csv', index_col='data', parse_dates=True)
df.index.freq='MS'

'''
# Plot de grafico para analise dos dados
res = seasonal_decompose(df['consumo'])
res.plot()
plt.show()
'''

# Print para determinar o tamanho da tabela e consequentemente o tamanho de treino e teste
print(len(df))

# Treino e Teste
train = df.iloc[:204]
test = df.iloc[204:]

# Rescreve os dados para um intervalo de 0 a 1
scaler = MinMaxScaler()
scaler.fit(train)
scaled_train = scaler.transform(train)
```

```
scaled_test = scaler.transform(test)

n_input = 60 #12
n_features = 1
generator = TimeseriesGenerator(scaled_train, scaled_train, n_input, batch_size)
x,y = generator[0]
print("x: {}".format(x))
print("y: {}".format(y))

# Define o modelo
model = Sequential()
model.add(LSTM(100, activation='relu', input_shape=(n_input, n_features)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')

#print(model.summary())

model.fit(generator, epochs=40)

# Plota o grafico de perdas para definir o epochs ideal
loss_per_epoch = model.history.history['loss']
#plt.plot(range(len(loss_per_epoch)), loss_per_epoch)
#plt.show()

last_train_batch = scaled_train[:60] #12
last_train_batch = last_train_batch.reshape(1, n_input, n_features)

#print(model.predict(last_train_batch))

#print(scaled_test[0])

test_predictions = []

first_eval_batch = scaled_train[-n_input:]
current_batch = first_eval_batch.reshape((1, n_input, n_features))

for i in range(len(test)):
    current_pred = model.predict(current_batch)[0]
    test_predictions.append(current_pred)
    current_batch = np.append(current_batch[:,1:,:], [[current_pred]], axis=1)

# Retorna os dados para os valores reais
true_predictions = scaler.inverse_transform(test_predictions)

df2 = pd.DataFrame(true_predictions, index=range(0,12), columns=['Consumo'])

test = test.rename_axis('data').reset_index() # Reseta o index
```

```
test[‘forecast’] = df2[‘Consumo’]
test = test.set_index(‘data’)

test.plot(figsize=(12,6))
plt.show()

# Cria a coluna com o erro percentual
test[‘MAPE’] = (100 * (test[‘consumo’] - test[‘forecast’])) / test[‘consumo’]

print(test)
```

