MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA BAIANO -
***CAMPUS* CATU**
CURSO TECNÓLOGO EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

TRABALHO DE CONCLUSÃO DE CURSO

LUIZ VITOR SOARES SOUZA

UMA ABORDAGEM DE GERENCIAMENTO DE ENERGIA EM UM NÓ SENSOR DE UMA REDE DE SENSORES SEM FIO

CATU-BAHIA
2025

LUIZ VITOR SOARES SOUZA

UMA ABORDAGEM DE GERENCIAMENTO DE ENERGIA EM UM NÓ SENSOR DE UMA REDE DE SENSORES SEM FIO

Trabalho de Conclusão de Curso apresentado ao curso de Análise e Desenvolvimento de Sistemas como requisito parcial para à obtenção do grau de Tecnólogo em Análise e Desenvolvimento de Sistemas pelo Instituto Federal de Educação, Ciência e Tecnologia Baiano *Campus* Catu.

Orientador(a): Prof. Dr. Társio Ribeiro Cavalcante

CATU - BA

2025

Ficha catalográfica elaborada pelo Bibliotecário Ricardo Santos do Carmo Reis - CRB – 5ª / 1649

Souza, Luiz Vitor Soares

729a Uma abordagem de gerenciamento de energia em um nó sensor de uma rede de sensores sem fio/ Luiz Vitor Soares Souza.- Catu, BA, 2025.

67 p.; il.: color.

Inclui bibliografia.

Trabalho de Conclusão de Curso (Graduação em Tecnologia em Análise e Desenvolvimento de Sistemas) – Instituto Federal de Educação, Ciência e Tecnologia Baiano – Campus Catu.

Orientador: Prof. Dr. Társio Ribeiro Cavalcante.

1. Redes de sensores sem fio. 2. Gerenciamento de energia. 3. Consumo energética. 4. Autonomia de nós sensores. I. Instituto Federal de Educação, Ciência e Tecnologia Baiano. II. Cavalcante, Társio Ribeiro (Orient.). III. Título.

CDU: 004.78

Agradecimentos

Primeiramente, agradeço aos meus pais pelo apoio, pela paciência e pelo incentivo ao longo de toda esta jornada. Estendo meus agradecimentos aos demais familiares, ao meu orientador, e a todos que, de alguma forma, contribuíram para a realização deste trabalho.

RESUMO

O trabalho investiga como prolongar a autonomia de nós em Redes de Sensores Sem Fio, cenário comum em aplicações que dependem de bateria e operam longe de manutenção frequente. Busca-se construir e validar um modelo de gerenciamento que ajuste o comportamento do nó conforme o nível de carga, alternando entre modos de operação para reduzir gastos desnecessários sem perder dados relevantes. Como metodologia, foi adotada a *Design Science Research, ocasião em que* partiu-se da definição do problema e dos requisitos, propôs-se a solução, desenvolveu-se o artefato, culminando, por fim, no processo de avaliação da solução. O protótipo utiliza um nó com sensores ambientais e comunicação via Wi-Fi, que envia leituras para um servidor PHP com MySQL hospedado na WEB; uma interface web em HTML, CSS e JavaScript permite acompanhar as medições em tempo quase real e forçar a troca de modo quando necessário. Foi observada uma queda mensurável no consumo e um aumento do tempo de vida do nó quando comparado a um funcionamento sem política adaptativa, além de estabilidade operacional mesmo em níveis baixos de bateria. A solução possibilitou concluir que uma política simples, implementável com baixo custo e recursos amplamente disponíveis, é suficiente para estender a autonomia e dar previsibilidade ao comportamento do nó, servindo de base para implantações maiores e para futuras extensões com ajustes dinâmicos mais finos.

Palavras-chave: Redes de Sensores Sem Fio; Gerenciamento de Energia; Consumo Energético; Autonomia de Nós Sensores.

ABSTRACT

The work investigates how to extend the autonomy of nodes in Wireless Sensor Networks, a common scenario in applications that rely on batteries and operate far from frequent maintenance. The goal is to design and validate a management model that adjusts node behavior according to battery level, switching between operation modes to reduce unnecessary consumption without losing relevant data. The adopted methodology was Design Science Research, beginning with the definition of the problem and requirements, followed by proposing the solution, developing the artifact, and finally evaluating the results. The prototype uses a node equipped with environmental sensors and Wi-Fi communication, sending readings to a PHP server with MySQL hosted on the web; a web interface built with HTML, CSS, and JavaScript allows near real-time monitoring of the measurements and manual mode switching when necessary. A measurable reduction in power consumption and an increase in node lifetime were observed compared to operation without an adaptive policy, as well as stable performance even at low battery levels. The solution demonstrated that a simple, low-cost policy using widely available resources is sufficient to extend autonomy and provide predictable node behavior, serving as a foundation for larger deployments and future extensions with more refined dynamic adjustments.

Keywords: Wireless Sensor Networks; Energy Management; Power Consumption; Sensor Node Autonomy.

Lista de Figuras

Figura 1 - Esquema básico de funcionamento de um transdutor. 14
Figura 2 - Modelo básico de uma rede de sensores. 16
Figura 3 - Exemplo da arquitetura básica de um nó sensor. 17
Figura 4- Relacionamento entre serviços, funções e modelos na arquitetura Manna. 22
Figura 5 – Representação do Duty Cycling em um nó sensor. 26
Figura 6 - Resultados do trabalho Sousa et al. (2007) 31
Figura 7. Processo metodológico para DSR. 33
Figura 8 - Interface Web 44
Figura 9 - Arquitetura geral do sistema 45
Figura 10 - Seleção de modo de operação na interface web 51

Lista de Quadros

Quadro 1 – Estados de operação, faixas de tensão (emulação 12 V) e função 35
Quadro 2 – Parâmetros de histerese (limiares de entrada e retorno) para o nó sensor a 12 V 37
Quadro 3 - Sensores do nó para simulação 40
Quadro 4 – Sensores ativos conforme a faixa de operação (1 ligado, 0 desligado) 47
Quadro 5 – Consumo energético e autonomia por faixa 48

Sumário

1. Introdução 9
1.1. Contextualização 9
1.2. Problema 11
1.3. Hipótese 11
1.4. Justificativa 11
1.5. Objetivo Geral 12
1.6. Objetivos Específicos 12
1.7 Estrutura do trabalho 12
2. Fundamentação teórica 13
2.1 Sensores 13
2.2 Rede de sensores sem fio 14
2.3 Nó sensor 16
2.4 Desafios energéticos das RSSF 18
2.5 Consumo energético em RSSF 19
2.6 Arquitetura MANNA 21
2.7 Gerenciamento Baseado em Energia Residual 22
2.8 Gerenciamento de energia em RSSF 23
2.9 Duty Cycling: Ciclos de Atividade Intermitentes 25
3. Trabalhos correlatos 27
3.1 Metodologia da busca por trabalhos correlatos 27
3.2 Revisão dos trabalhos correlatos 28
4. Metodologia 31
4.1 Tipo de Pesquisa 32
4.2 Etapas de pesquisa da DSR e o que foi realizado no trabalho 33
4.3 Método de Desenvolvimento do artefato 34
4.4 Procedimentos de teste 37
5. Artefato: projeto, implementação e modelagem 39
5.1 Hardware 39
5.2 Software 40
5.2.1 Servidor e interface Web 42
5.2 Arquitetura geral 43
6. Resultado e Discussões 45
6.1 Prioridade dos Sensores 45
6.2 Consumo energético estimado 46
6.3 Comparativo de autonomia e comportamento adaptativo 47
6.4 Validação da mudança de modo via interface web 49
6.5 Discussão dos resultados 50
7. Conclusão 51
8. Referências 52
APÊNDICE A – Código-fonte do Arduino (Módulo Sensor e Controle de Energia) 55
APÊNDICE B – Código-fonte do ESP8266 (Comunicação e Interface Web) 61
APÊNDICE C – Detalhamento do Hardware e Mapeamento de Pinos 64

1. Introdução

1.1. Contextualização

O avanço constante na área de microprocessadores, o desenvolvimento de novos materiais de sensoriamento, o progresso em microssistemas eletromecânicos (conhecidos como MEMS - *Micro Electro-Mechanical Systems*) e as inovações na comunicação sem fio têm desempenhado um papel crucial na promoção do desenvolvimento e adoção de sensores "inteligentes" em diversos domínios relacionados a processos físicos, químicos, biológicos e outras áreas afins. Um cenário comum envolve a integração de vários sensores em um único chip, onde a lógica do circuito integrado controla esses sensores, e uma interface de comunicação sem fio possibilita a transferência de dados, formando assim uma Rede de Sensores Sem Fio (RSSF) (LOUREIRO *et al.* 2003).

Uma RSSF representa um sistema computacional projetado para monitorar diversos fenômenos físicos do ambiente, como por exemplo, temperatura, umidade ou vibração. Uma RSSF consiste em um conjunto de dispositivos interconectados, cada um equipado com sensores específicos, assumindo a função de nós individuais na rede. Esses dispositivos se comunicam por meio de tecnologias de comunicação sem fio, criando uma estrutura dinâmica de coleta e compartilhamento de dados (FOSTER, 2016).

Este tipo de rede pode ser formada por centenas ou milhares de dispositivos autônomos que tendem a ser projetados com pequenas dimensões chamadas nós sensores (RUIZ *et al*., 2004). Os principais elementos de um nó sensor consistem em um transceptor, responsável por enviar e receber dados sem fio entre os nós sensores e outros dispositivos da rede para comunicação, uma fonte de energia, uma unidade de sensoriamento, memória e um processador. O aspecto lógico fundamental de um nó sensor é representado pelo seu *software* (LOUREIRO *et al.*, 2003). Os nós individualmente possuem pouca capacidade computacional e de energia, mas um esforço colaborativo entre os mesmos permite a realização de uma tarefa mais complexa (RUIZ *et al*., 2004).

Do ponto de vista científico, as RSSFs apresentam diversos desafios que ainda não foram completamente estudados ou que estão em estágios iniciais de exploração. Um dos principais desafios é a questão da manutenção dessas redes.

Em geral, os nós sensores possuem mobilidade limitada ou nula e sua fonte de energia, geralmente baterias, está embutida em sua estrutura (CERQUEIRA, COSTA, 2019). Quando a vida útil das baterias chega ao fim, a localização dos nós de RSSF pode gerar dificuldades no acesso humano para atividades de manutenção, além de aumentar os custos envolvidos. Portanto, a limitação energética representa um dos principais desafios enfrentados pelas RSSF.

Neste trabalho de conclusão de curso o intuito é prolongar a autonomia e preservar serviços essenciais em redes de sensores alimentadas por baterias, reduzindo intervenções humanas de manutenção. Para isso, no nó sensor será implementado um mecanismo de gerenciamento de energia que mede a tensão da bateria, estimando o estado de carga e controlando o acionamento das portas do arduino por dois modos: automático, via algoritmo com limiares de tensão, modo economia, e manual, por meio de uma interface web móvel que exibe em tempo real a porcentagem de bateria e permite ligar/desligar portas específicas. A eficácia será avaliada por métricas como consumo médio, autonomia do nó e disponibilidade, comparando o desempenho com e sem o algoritmo proposto.

A implementação de um sistema de gerenciamento de energia em uma RSSF é importante principalmente pela sua localização, já que na maioria dos casos elas são implantadas em locais de difícil acesso, então ter esse controle de energia ajuda a maximizar a vida útil dos sensores e otimizar sua eficiência energética. Técnicas e algoritmos de gerenciamento de energia têm sido amplamente estudados e desenvolvidos para equilibrar o consumo energético com o desempenho da rede (AKYILDIZ *et al*., 2002; Heinzelman *et al*., 2000). No contexto deste trabalho, a abordagem proposta foca na monitorização e no controle da capacidade energética das baterias que alimentam os sensores e demais componentes do nó sensor. A estratégia envolve medir a tensão da rede através de um sensor de tensão e a avaliação do nível de carga da bateria através de um algoritmo que irá ser implementado no microcontrolador do nó sensor. Essa abordagem permite uma gestão mais precisa e adaptativa da energia, assegurando a continuidade da operação dos sensores e minimizando interrupções, o que é essencial para a manutenção da rede a longo prazo.

1.2. Problema

Como gerenciar o consumo energético de um nó sensor em uma RSSF para maximizar sua autonomia operacional?

1.3. Hipótese

A integração de medição de tensão e controle automático e manual das portas do arduino, por meio de algoritmo de gerenciamento energético, pode aumentar a autonomia operacional do nó sensor em relação ao funcionamento sem gerenciamento.

1.4. Justificativa

A crescente demanda por RSSFs têm revolucionado a forma como interagimos com nosso ambiente, permitindo monitoramento, coleta e análise de dados em uma escala sem precedentes. No entanto, esse avanço tecnológico também apresenta desafios críticos, dos quais o gerenciamento de energia em nós sensores se destaca como uma questão primordial a ser abordada.

A justificativa deste projeto de trabalho de conclusão de curso reside na necessidade de melhorar a eficiência energética de um nó de RSSF utilizando dispositivos de hardware para monitorar a energia da rede e um algoritmo de gerenciamento energético que será implantado no microcontrolador do nó. Um desafio cuja solução é vital para o pleno aproveitamento dessa rede. A operação de nós sensores frequentemente ocorre em ambientes inóspitos e de difícil acesso, onde a substituição constante de baterias ou manutenção torna-se impraticável e custosa. Além disso, em áreas críticas, como monitoramento ambiental, controle industrial e assistência médica, qualquer interrupção nas operações pode ter consequências graves.

O gerenciamento de energia surge como uma resposta a essas limitações e desafios. A eficácia dessa abordagem pode resultar em um aumento substancial na autonomia dos nós sensores, garantindo operações ininterruptas e reduzindo significativamente os custos associados à manutenção e substituição. Além disso, a capacidade de adaptação a diferentes cenários de energia pode permitir que a

RSSF continue a operar com eficiência em situações de crise ou com recursos energéticos limitados.

Além disso, o avanço no gerenciamento de energia em RSSFs tem um impacto significativo no campo da pesquisa em sistemas distribuídos, redes de comunicação e Internet das Coisas (IoT). Estudos mostram que a eficiência energética é um dos principais fatores que determinam a viabilidade e a longevidade desses sistemas, influenciando diretamente a qualidade da comunicação e o desempenho das aplicações IoT (AKYILDIZ et al., 2002; RAZZAQUE; HARJANI, 2017). Contribuir para a compreensão e solução desse desafio tecnológico em constante evolução é, portanto, não apenas uma resposta às necessidades atuais, mas também um investimento no progresso científico.

1.5. Objetivo Geral

Desenvolver um mecanismo de gerenciamento de energia para nós sensores de uma RSSF que possibilite otimizar o consumo energético e aumentar a autonomia do dispositivo na rede.

1.6. Objetivos Específicos

- Implementar ao nó sensor um sensor de tensão calibrado para estimar o estado de carga.
- Projetar o controle de portas do arduino com limiares através de um algoritmo definindo os estados da bateria em normal, economia, crítico, evitando comutação frequente e priorizando serviços essenciais.
- Implementar modo manual via comando do usuário e lógica de prioridade entre manual e automático, com estados seguros em baixa tensão.
- Desenvolver UI que mostra a porcentagem da bateria e status das portas em tempo quase real, permitindo ligar ou desligar portas e alternar entre modos, com feedback de confirmação, essa UI também permite que o usuário defina serviços essenciais.
- Executar testes comparativos com e sem o algoritmo de gerenciamento, quantificando o ganho de autonomia.

1.7 Estrutura do trabalho

Este trabalho está estruturado da seguinte forma: o Capítulo 2 apresenta a fundamentação teórica, delineando os conceitos e componentes das RSSF e os mecanismos de gerenciamento energético; o Capítulo 3 reúne os trabalhos correlatos e descreve a metodologia de busca adotada; o Capítulo 4 detalha a metodologia (Design Science Research), incluindo tipo de pesquisa, etapas da DSR, método de desenvolvimento do artefato e procedimentos de teste; o Capítulo 5 descreve o artefato proposto, projeto, implementação e modelagem, abrangendo *hardware*, *software*, servidor e interface web e a arquitetura geral; o Capítulo 6 apresenta os resultados e discussões; o Capítulo 7 traz a conclusão e indica trabalhos futuros; por fim, o Capítulo 8 lista as referências utilizadas.

2. Fundamentação teórica

Esta seção apresenta os principais conceitos e fundamentos teóricos que embasam o desenvolvimento deste trabalho, oferecendo uma visão ampla sobre os elementos que compõem as RSSF e os mecanismos de gerenciamento energético nelas aplicados, como os sensores e sua função essencial na coleta de dados ambientais, as estruturas das redes de sensores sem fio e a composição dos nós sensores, os desafios energéticos enfrentados por essas redes e os fatores que influenciam o consumo energético dos nós. Em seguida, são exploradas algumas abordagens e modelos relevantes, como a arquitetura MANNA (*Management Architecture for Wireless Sensor Networks*) e o conceito de *duty cycling*, técnica amplamente utilizada para reduzir o consumo energético por meio da alternância entre modos de atividade e repouso.

2.1 Sensores

Sensores são os principais dispositivos em sistemas de monitoramento, especialmente em Redes de Sensores Sem Fio (RSSF), pois permitem a coleta de dados ambientais em tempo real. Segundo Balbinot e Brusamarello (2011), sensores são capazes de captar grandezas físicas, químicas ou biológicas e convertê-las em sinais elétricos, os quais podem ser processados e interpretados por sistemas

computacionais. Cada sensor possui uma arquitetura própria, composta por elementos como o elemento sensor, responsável pela interação com o ambiente, o transdutor, que converte essa informação em um sinal elétrico e, em alguns casos, um circuito de condicionamento de sinal, que ajusta a saída para posterior interpretação (TEIXEIRA, 2023; COUNCIL, 1997). A Figura 1 ilustra o esquema básico de um sensor. Em uma RSSF, sensores são incorporados a nós sensores e operam de forma cooperativa, sendo capazes de realizar sensoriamento distribuído de fenômenos físicos como temperatura, umidade, pressão, vibração, presença de gases, entre outros. Essa atuação cooperativa permite que mesmo sensores de baixo custo e capacidade limitada possam realizar tarefas complexas quando conectados em rede (LOUREIRO *et al*. 2003).

Figura 1 - Esquema básico de funcionamento de um transdutor.

Fonte: Brasil Escola. *Transdutor*. (s.d.).

A evolução tecnológica permitiu o desenvolvimento de sensores mais precisos, miniaturizados e com maior eficiência energética. MEMS (Micro-Electro-Mechanical Systems) são um exemplo dessa inovação: sensores miniaturizados que combinam componentes mecânicos e eletrônicos em um único chip, permitindo medições mais rápidas e com menor consumo (FORSTER, 2016). A escolha do sensor ideal depende diretamente do tipo de grandeza a ser monitorada, da precisão requerida, do ambiente de instalação e das restrições energéticas do nó sensor. Como aponta Yick *et al*. (2008), sensores com baixo consumo de energia e alta durabilidade são preferidos em aplicações onde a manutenção é difícil ou inviável. Além disso, é importante considerar o tipo de sinal gerado: sensores analógicos fornecem uma variação contínua de tensão proporcional à grandeza medida, enquanto sensores digitais realizam uma conversão interna e fornecem o valor já discretizado ao sistema. Sistemas embarcados como o Arduino e o ESP32

permitem a leitura de ambos os tipos por meio de suas entradas analógicas e digitais, adaptando-se a diferentes tecnologias sensoriais (POLASTRE *et al*., 2004).

2.2 Rede de sensores sem fio

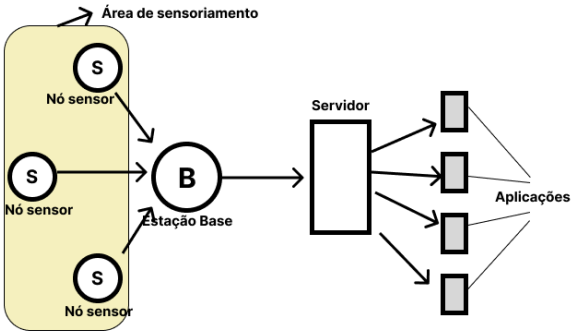
As RSSF caracterizam-se por sistemas distribuídos nos quais os dispositivos operam de forma autônoma. Dentro desse contexto, os componentes sensoriais adquirem uma função multifacetada, indo além da mera coleta de dados ambientais, para também desempenhar papéis de controle e resposta. Essa capacidade intrínseca possibilita que os sensores exerçam influência direta sobre os atuadores, por meio da transmissão de sinais de controle em consonância com os processos vigentes (LOUREIRO *et al*., 2003).

A configuração de uma RSSF geralmente envolve a utilização de diversos nós sensores, cada um equipado com sensores específicos para detectar informações relevantes do ambiente, como temperatura, umidade, pressão, vibração, luminosidade, entre outros. A rede é formada pela interconexão desses nós por meio de tecnologias de comunicação sem fio, como rádio frequência, ZigBee, LoRa, BLE, ou até protocolos Wi-Fi de baixo consumo (FOSTER, 2016; YICK *et al*., 2008). Além disso, em uma RSSF também é importante destacar o papel da estação base, que atua como o principal ponto de coleta e interface entre a rede de sensores e o sistema externo de processamento e análise de dados.

A estação base é responsável por receber os dados transmitidos pelos nós sensores, tratar eles e encaminhá-los para um servidor local ou remoto, onde serão processados e visualizados e enfim utilizados para algum fim. Segundo Forster (2016), a estação base geralmente possui maior capacidade computacional e energética em relação aos nós sensores, e é equipada com módulos de comunicação mais potentes (como Wi-Fi, LoRa, ou mesmo conexão cabeada Ethernet), além de poder incluir microcontroladores, microprocessadores como Raspberry Pi ou placas embarcadas, e interfaces para comunicação serial ou USB. Em algumas RSSF, através da estação base por meio de alguma interface web ou aplicativos conectados a ela também é possível enviar comandos de controle ou reconfiguração para os nós sensores da rede, isso permite alterar intervalos de amostragem, ativação ou desativação de sensores, ou comandos para entrar em modos de economia de energia (YICK *et al*., 2008).

Outra função importante da estação base é minimizar o consumo de energia de cada nó sensor, já que ela centraliza as informações captadas pelos sensores espalhados na rede, o que permite reduzir o tráfego entre os nós já que nem todos precisam transmitir diretamente para a internet, a estação que assume esse papel. De acordo com LOUREIRO *et al.* (2003), essa arquitetura hierárquica, onde a estação base atua como concentrador de dados (*data sink*), é essencial para a escalabilidade e eficiência das redes de sensores sem fio. A Figura 2 exemplifica uma estrutura básica dos principais componentes de uma rede de sensores sem fio.

Figura 2 - Modelo básico de uma rede de sensores.



Fonte: Do autor

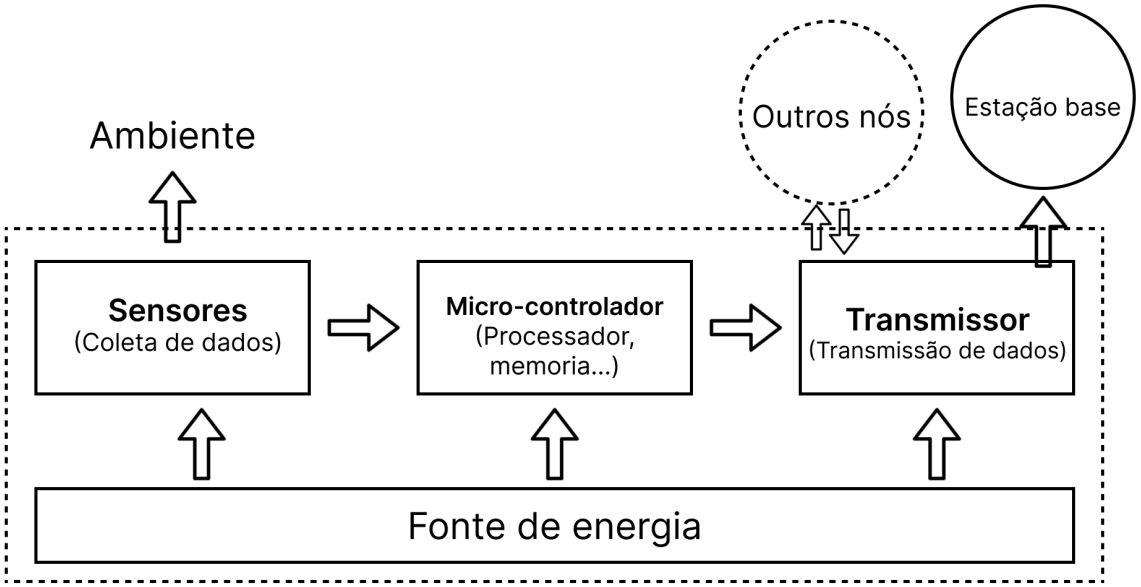
2.3 Nó sensor

Os nós sensores são, pequenas unidades computacionais embarcadas que atuam de forma autônomas e que em conjunto formam a Rede de sensores sem fio, seu papel é detectar fenômenos físicos ou químicos, processar estes dados coletados localmente e transmiti-los para uma estação base ou até mesmo para algum outro nó da rede. (FOSTER, 2016).

Para o seu funcionamento os nós sensores precisam de alguns elementos que são fundamentais. O processador que pode ser um microcontrolador como um Arduino ou o ESP32, ele é responsável por executar as rotinas de coleta de dados, filtragem e tomada de decisão com base nestes dados captados. O transmissor, que pode operar em protocolos como ZigBee, LoRa ou Wi-Fi, é o componente responsável pela comunicação entre os nós e a estação base. Os sensores são responsáveis pela detecção das grandezas ambientais (temperatura, umidade, luz, gases, entre outros). Por fim a fonte de energia, este o maior fator limitante da

operação contínua de um nó sensor, geralmente a fonte é uma bateria de lítio ou supercapacitor que alimenta todos os demais componentes compostos no nó (LOUREIRO *et al* 2003). A Figura 3 mostra um exemplo de uma arquitetura de um nó sensor de uma RSSF.

Figura 3 - Exemplo da arquitetura básica de um nó sensor.



Fonte: Adaptado de Karl e Willig (2007).

Segundo Forster (2016), uma das principais características de um nó sensor é sua a sua limitação recursos, então o seu baixo poder de processamento, a pouca memória disponível e capacidade energética reduzida pode ser vista como algo positivo já que isso proporciona a redução de custos e a viabilidade da implantação em larga escala. Com isso, uma rede pode ser composta por centenas ou milhares de nós de baixo custo, isso permite com que cada nó seja projetado para desempenhar apenas funções específicas com o mínimo consumo de energia possível, otimizando assim seu tempo de vida útil. Esta simplicidade de cada nó é compensada pela inteligência coletiva da rede, mesmo com recursos limitados, a colaboração entre nós permite realizar tarefas complexas como o monitoramento distribuído de grandes áreas, tomada de decisão descentralizada e reconfiguração automática da rede em caso de falhas.

Geralmente, as RSSFs são implementadas com nós sensores estáticos, ou seja, dispositivos fixos instalados em locais estratégicos de acordo com o objetivo da

aplicação. No entanto, existem também redes com nós móveis, em que os sensores são embarcados em veículos terrestres, aquáticos ou aéreos, como drones, robôs autônomos ou até mesmo em animais silvestres para fins de monitoramento ambiental ou rastreamento comportamental.

De acordo com García-Hernández *et al.* (2007), os nós sensores possuem diversas áreas de aplicação, devido à sua flexibilidade, baixo custo e capacidade de operar em ambientes diversos. Dentre as principais aplicações, destacam-se:

- **Área industrial**: utilizados no monitoramento e controle automatizado de equipamentos e processos, contribuindo para a manutenção preditiva e o aumento da eficiência operacional;
- **Agricultura**: empregados no monitoramento de condições ambientais como umidade do solo, temperatura, luminosidade e índice pluviométrico.
- **Segurança pública**: empregados na detecção antecipada de situações de risco, como áreas suscetíveis a desastres naturais (enchentes, deslizamentos, incêndios florestais);
- **Situações emergenciais**: fundamentais para a detecção de incêndios, vazamentos de gás, condições ambientais críticas e outros eventos que exijam resposta rápida.

2.4 Desafios energéticos das RSSF

As RSSF são amplamente utilizadas em diversas aplicações, como monitoramento ambiental, agricultura de precisão, segurança, saúde e automação industrial. No entanto, apesar de seus benefícios, as RSSFs enfrentam diversos desafios técnicos que ainda precisam ser superados para garantir sua operação eficiente, confiável e sustentável ao longo do tempo. De acordo com o relatório da Workshop on Fundamental Research in Networking, patrocinado pela National Science Foundation (RUIZ *et al.*, 2004), muitos desses desafios ainda não foram completamente estudados e solucionados. Entre os principais obstáculos, destacam-se: a operação em ambientes hostis, a manutenção da conectividade entre os nós sensores, a eficiência no controle e roteamento da rede, a escalabilidade e a segurança na comunicação (TIWARI *et al*., 2015).

No entanto, um dos problemas mais críticos para a longevidade dessas redes é a eficiência energética. O tempo de vida útil de um nó sensor está diretamente

relacionado à durabilidade de sua fonte de energia, geralmente é uma bateria com capacidade limitada. Essa dependência energética tem impactos no desempenho da rede. Como cada nó sensor em RSSFs com topologia distribuída pode exercer funções de sensoriamento, processamento e encaminhamento de dados de outros nós, a falha de algum nó por esgotamento de energia pode comprometer as rotas previamente estabelecidas, causando a interrupção no fluxo de dados e obrigando a rede a realizar reconfigurações dinâmicas no roteamento ou até mesmo o interrompimento operacional de toda a RSSF comprometendo sua cobertura e confiabilidade (Yick *et al*., 2008).

Ainda segundo Heinzelman *et al.* (2000), a maior parte da energia consumida por um nó sensor não está associada ao processamento interno, mas sim às operações de comunicação sem fio, como a transmissão e recepção de pacotes de dados. Isso ocorre porque manter o transmissor ativo exige um consumo contínuo significativo das baterias. Além disso, o envio de pacotes redundantes, as falhas na comunicação, as retransmissões e o congestionamento da rede contribuem para o desperdício energético, reduzindo a eficiência geral do sistema. Em muitos casos, a substituição ou recarga da bateria não é uma opção viável, seja por restrições físicas de acesso, seja por custos operacionais elevados, especialmente em aplicações remotas, subterrâneas ou subaquáticas (TIWARI *et al*., 2015).

Outro fator importante a considerar é que, em muitas aplicações, os nós sensores precisam operar continuamente ou em intervalos regulares, o que exige um equilíbrio cuidadoso entre a frequência de amostragem, o volume de dados transmitidos e o nível de energia disponível. Técnicas como *duty cycling*, compressão de dados, roteamento energético eficiente e protocolos de comunicação de baixo consumo têm sido amplamente estudadas como alternativas para minimizar esse impacto (POLASTRE *et al.*, 2004; YE, HEIDEMANN & ESTRIN, 2002).

2.5 Consumo energético em RSSF

Os desafios relacionados ao consumo energético em Redes de Sensores Sem Fio (RSSF) devem ser levados em consideração ainda durante a fase de elaboração e implementação dos nós sensores, já que esses dispositivos, na maioria das vezes, são instalados em áreas remotas, de difícil acesso ou até mesmo inóspitas. Essa limitação geográfica dificulta qualquer serviço relacionado à

manutenção dos nós, principalmente as trocas ou recarga das baterias da fonte de alimentação, o que torna essa tarefa custosa, arriscada e muitas vezes inviável.

O consumo energético de um nó sensor está diretamente ligado a diversos fatores, como o tipo de aplicação, o número e a natureza dos sensores embarcados, o protocolo de comunicação utilizado, a frequência de coleta e transmissão de dados, além da capacidade de processamento local. Cada um desses elementos influencia a taxa de consumo de energia do nó, o que, por sua vez, afeta significativamente a autonomia e o tempo de vida útil da rede como um todo (Heinzelman *et al.*, 2000; Polastre *et al.*, 2004). Nas aplicações que exigem sensoriamento contínuo e transmissão em tempo real geralmente demandam mais energia do que aquelas baseadas em coletas periódicas ou em eventos específicos. Da mesma forma, protocolos de comunicação com retransmissões frequentes ou baixa eficiência de roteamento contribuem para o aumento do consumo energético, especialmente quando os nós precisam operar em modo de escuta constante. Por isso é importante considerar estratégias de otimização energética já no projeto do nó sensor, na escolha dos componentes de hardware, nas rotinas de software embarcado e também definir algoritmos de gerenciamento. Isso garante que a rede tenha maior autonomia operacional aumentando assim sua sustentabilidade e a viabilidade em longo prazo, especialmente em ambientes onde a intervenção humana é limitada.

Uma possível abordagem para enfrentar estes desafios energéticos enfrentados por RSSF é a implementação de fontes de energia renovável integradas aos nós sensores da rede. Essa estratégia visa ampliar a autonomia operacional da rede, reduzindo a necessidade de manutenção constante e diminuindo a dependência exclusiva das baterias embutidas na rede, que possuem vida útil limitada e, frequentemente, não podem ser substituídas com facilidade em ambientes remotos (AKYILDIZ *et al.*, 2002). Ao incorporar fontes de energia sustentáveis, como painéis solares fotovoltaicos ou microturbinas eólicas, os nós sensores da rede podem coletar energia diretamente do ambiente em que estão inseridos, permitindo um funcionamento mais contínuo e confiável.

Segundo Goldemberg e Lucon (2007), o uso de recursos naturais como o sol e o vento para geração de energia é essencial para o desenvolvimento de tecnologias mais sustentáveis e de baixo impacto ambiental. Essa alternativa não apenas minimiza a necessidade de intervenções humanas para substituição de

baterias, como também simplifica a gestão e manutenção da rede. Além disso, favorece uma abordagem mais ecológica e resiliente, especialmente em aplicações de longo prazo, como monitoramento ambiental, agricultura de precisão e estações meteorológicas remotas. Vullers *et al*. (2009) destaca ainda que a utilização de sistemas híbridos, por exemplo, energia solar e eólica, podem compensar a intermitência de cada fonte individual, garantindo maior estabilidade na alimentação dos nós sensores.

2.6 Arquitetura MANNA

Um dos primeiros estudos a abordar o gerenciamento sistemático em RSSF foi o trabalho de Ruiz *et al.* (2003), que propôs a arquitetura MANNA (*Management Architecture for Wireless Sensor Networks*). A MANNA foi desenvolvida com o propósito de atender às necessidades específicas de gerenciamento em redes de sensores, oferecendo uma estrutura modular, escalável e independente da tecnologia de comunicação utilizada, o que a torna aplicável a uma ampla variedade de cenários e topologias de rede.

A arquitetura propõe a organização do gerenciamento em quatro domínios principais:

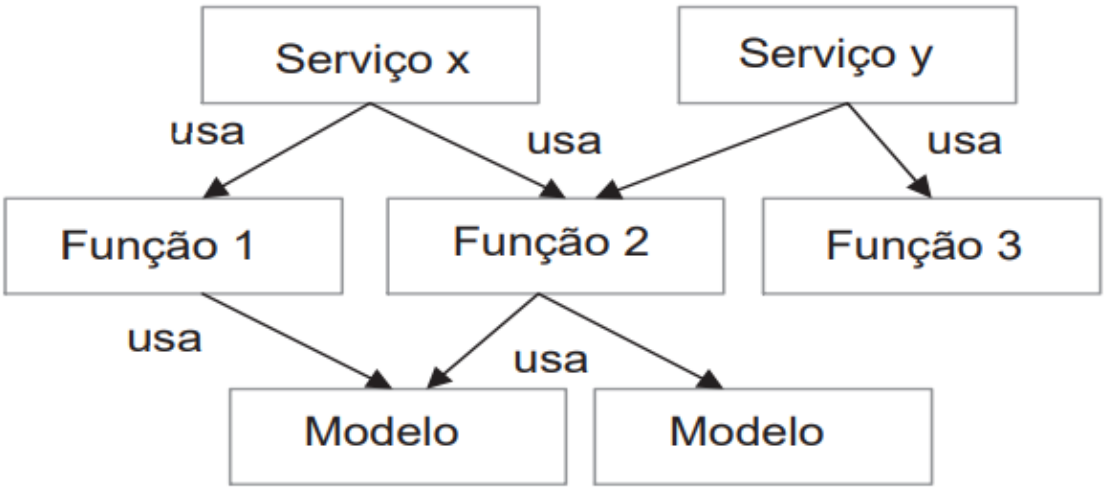
- Gerenciamento de negócios (relacionado aos objetivos da aplicação);
- Gerenciamento de serviços (responsável por coordenar os serviços prestados pela rede);
- Gerenciamento da rede (voltado para a conectividade, topologia e comunicação entre os nós);
- Gerenciamento dos elementos da rede (que lida diretamente com os dispositivos físicos, como os nós sensores).

Cada serviço de gerenciamento é composto por funções específicas, que são executadas com base em informações contextuais extraídas da própria rede. Essas funções se apoiam em mapas, que são modelos internos representando o estado atual da rede, como por exemplo:

- Mapa de energia: indica a energia residual dos nós sensores;
  Mapa de topologia: exibe a estrutura da rede e sua conectividade atual;
- Mapa de desempenho: mostra métricas como perda de pacotes, atraso, etc.

A utilização desses mapas permitem que a MANNA realize ações de controle e adaptação automática, como desligar sensores não essenciais, reorganizar rotas, redistribuir tarefas ou ajustar parâmetros operacionais com base no contexto da aplicação. Essa abordagem possibilita uma atuação proativa, reduzindo o impacto de falhas, prolongando a vida útil da rede e mantendo a qualidade do serviço prestado. A principal vantagem da MANNA reside na sua capacidade de abstrair a tecnologia subjacente da rede, permitindo o gerenciamento uniforme, independente do protocolo de comunicação adotado. Isso confere flexibilidade ao projeto, facilita a reutilização de estratégias em diferentes cenários e aumenta a autonomia da rede, fator essencial em aplicações remotas e de difícil manutenção (Ruiz, 2003). A Figura 4 exemplifica o funcionamento da arquitetura MANNA, evidenciando a relação entre os serviços de gerenciamento, as funções associadas e os mapas ou modelos de estado que orientam as ações de controle na rede.

Figura 4- Relacionamento entre serviços, funções e modelos na arquitetura Manna.



Fonte: Ruiz (2003).

2.7 Gerenciamento Baseado em Energia Residual

O gerenciamento baseado em energia residual é uma abordagem dinâmica e adaptativa, utilizada em algumas RSSF com o objetivo de prolongar a vida útil da rede e otimizar o uso dos recursos energéticos disponíveis. Essa abordagem se baseia no monitoramento constante da quantidade de energia restante em cada nó sensor, utilizando essas informações como critério principal para a tomada de

decisões operacionais, como a priorização de nós ativos, reconfiguração de rotas e desligamento seletivo de sensores não essenciais (Heinzelman *et al*., 2000; TIWARI *et al.,* 2015).

Ao contrário de arquiteturas fixas, o gerenciamento baseado em energia residual é altamente responsivo ao estado atual da rede. À medida que a carga da bateria de um nó diminui, suas responsabilidades dentro da rede podem ser alteradas, permitindo que outros nós com maior reserva energética assumam as suas funções. Isso evita o esgotamento de nós críticos, como aqueles localizados próximos à estação base ou que atuam como roteadores intermediários, o que poderia causar falhas em cascata ou isolamento de segmentos da rede.

Para operacionalizar essa abordagem, pode-se utilizar diversas técnicas, como:

- *Duty cycling* adaptativo: ajusta os ciclos de sono/atividade com base na energia disponível;
- Roteamento sensível à energia: seleciona caminhos que passem por nós mais energizados;
- Agrupamento energético para balancear o tráfego e reduzir o consumo em nós sobrecarregados;
- Desligamento ou escalonamento de sensores de menor prioridade, dependendo da criticidade da medição e da energia restante.

Protocolos modernos de gerenciamento, como o LEACH (Low-Energy Adaptive Clustering Hierarchy), e sistemas embarcados como o TinyOS, já incorporam algoritmos que exploram esses princípios para otimizar o uso da energia em tempo real, minimizando o consumo desnecessário e melhorando a eficiência geral da rede (POLASTRE *et al*., 2004).

2.8 Gerenciamento de energia em RSSF

A adoção de uma abordagem sustentável para suprir a demanda energética em RSSF tem se tornado cada vez mais relevante, especialmente por meio da integração de fontes renováveis híbridas, como painéis solares, colheita de energia vibracional e microturbinas eólicas (AKYILDIZ *et al*., 2002; Vullers *et al*., 2009). No entanto, o simples uso dessas fontes não é suficiente para garantir a longevidade da rede. Sendo assim é preciso implementar estratégias eficientes de gerenciamento

energético, que otimizem o uso da energia disponível e assegurem a operação contínua e confiável dos nós sensores.

Ao se lidar com dispositivos alimentados por baterias, é essencial maximizar a eficiência energética, atuando tanto em nível de hardware quanto de software. Do ponto de vista do hardware, destacam-se práticas como a seleção de componentes de baixo consumo, utilização de reguladores eficientes e o emprego de modos de operação em *stand-by* ou *deep sleep*. Já no nível de software, são cruciais os protocolos de comunicação de baixo consumo, algoritmos de roteamento adaptativos e escalonamento inteligente de tarefas, que reduzem a atividade do nó nos momentos de menor criticidade (POLASTRE *et al.*, 2004; YICK *et al*., 2008).

Essas estratégias contribuem na redução da necessidade de manutenção e substituição de baterias, promovendo maior sustentabilidade ambiental e diminuindo o impacto ecológico da implantação de redes em larga escala (HEINZELMAN *et al.*, 2000; AL-KARAKI & KAMAL, 2004). Considerando que a conservação de energia é uma das restrições mais críticas no projeto de uma RSSF (AKYILDIZ *et al*., 2002), diversas soluções têm sido propostas para reduzir o consumo energético dos nós sensores. Entre essas soluções, destacam-se, técnicas de escalonamento de potência, protocolos de comunicação de baixo consumo, como S-MAC, T-MAC e B-MAC (POLASTRE *et al*., 2004), algoritmos de roteamento eficientes, como LEACH, PEGASIS e TEEN (HEINZELMAN *et al.*, 2000; Al-KARAKI & KAMAL, 2004) e a utilização de camadas intermediárias (*middleware*) capazes de gerenciar dinamicamente a energia, permitindo que os nós entrem em modo de repouso de forma coordenada, sem comprometer a confiabilidade na coleta e transmissão dos dados (MOLLA & AHAMED, 2006).

Paralelamente, a aplicação de sistemas de colheita de energia *(energy harvesting*) tem se mostrado promissora, pois possibilita a captura, armazenamento e uso inteligente de energia ambiental. Técnicas como o gerenciamento adaptativo da bateria permitem que o nó sensor antecipe padrões de carga e descarga, ajustando dinamicamente a taxa de aquisição de dados e a frequência de transmissão de acordo com a energia disponível (Vullers *et al.*, 2009; Yick *et al.*, 2008). Dessa forma, mesmo em cenários adversos, os nós sensores podem permanecer operacionais, garantindo robustez, continuidade e confiabilidade à rede como um todo.

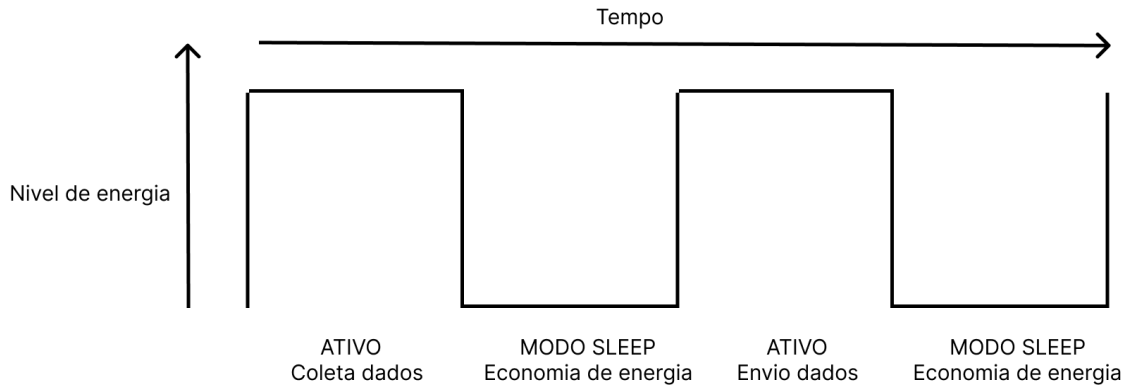
Outras técnicas de Gerenciamento Dinâmico de Energia (*Dynamic Power Management* – DPM) consistem em estratégias que visam reduzir o consumo energético dos nós sensores por meio do controle dinâmico dos estados operacionais de seus componentes internos. A ideia central é desativar, parcial ou totalmente, módulos específicos do nó (como rádio, sensores ou processador) quando não estiverem em uso, minimizando o desperdício de energia e, consequentemente, prolongando o tempo de vida útil da rede. Nessa abordagem, o nó sensor alterna entre estados de baixo consumo (*sleep mode*) e estados ativos (*active mode*) conforme a necessidade da aplicação. Após executar suas tarefas programadas (como sensoriamento ou transmissão de dados), o nó entra automaticamente em modo de repouso, retornando ao estado ativo somente em resposta a eventos relevantes ou em ciclos temporais definidos. Essa transição entre estados é realizada de forma autônoma e eficiente, permitindo economia energética sem comprometer a funcionalidade da rede.

2.9 Duty Cycling: Ciclos de Atividade Intermitentes

Uma das estratégias mais eficazes para reduzir o consumo energético em RSSF é o *duty cycling*, técnica que alterna períodos programados de atividade e de inatividade (modo de sono) dos nós sensores. Essa abordagem é particularmente eficaz porque a comunicação sem fio é responsável por grande parte do consumo de energia nesses dispositivos, em alguns casos, representando mais de 80% do total consumido, especialmente devido à escuta contínua do canal, que consome energia de forma semelhante à recepção de dados (Polastre *et al*., 2004; Ye, Heidemann & Estrin, 2002).

No *duty cycling*, o rádio do nó sensor é desligado durante os períodos em que ele não está transmitindo, recebendo ou escutando mensagens importantes, permanecendo em estado de repouso (*sleep mode*). Ele é então ativado somente nos momentos previamente definidos ou em resposta a eventos específicos. Isso reduz drasticamente o desperdício energético causado pela escuta ociosa (*idle listening*), que ocorre quando o nó permanece "ouvindo" o canal mesmo sem necessidade real de comunicação. A Figura 5 representa esse comportamento do *Duty Cycling* em um nó sensor.

Figura 5 – Representação do Duty Cycling em um nó sensor.



Fonte: Do autor.

Além da economia de energia, essa estratégia também melhora a eficiência operacional e aumenta a vida útil das baterias, sendo especialmente útil em aplicações que demandam longa duração, operação autônoma e onde a reposição energética é limitada ou inexistente.

Existem diferentes tipos de duty cycling, incluindo:

- ***Duty cycling* síncrono:** todos os nós seguem um mesmo cronograma de ativação e sono, exigindo sincronização precisa, porém com menor latência.
- ***Duty cycling* assíncrono:** cada nó possui seu próprio cronograma, acordando periodicamente para checar mensagens, o que reduz a necessidade de sincronização, mas pode aumentar o atraso na comunicação.
- ***Duty cycling* adaptativo:** o tempo de sono e atividade é ajustado dinamicamente com base em condições como tráfego da rede, energia residual ou prioridade da aplicação (Ye, Heidemann & Estrin, 2002).

3. Trabalhos correlatos

Esta seção apresenta os trabalhos correlatos que serviram de referência para o desenvolvimento desta pesquisa, destacando as principais contribuições, metodologias e resultados de estudos relevantes na área.

3.1 Metodologia da busca por trabalhos correlatos

Para buscar trabalhos correlatos sobre gerenciamento de energia em RSSF, foi realizada uma pesquisa bibliográfica de caráter exploratório e qualitativo. O objetivo era reunir e analisar estudos que apresentassem abordagens, modelos ou protocolos voltados à otimização do consumo energético em nós sensores, servindo como base para o desenvolvimento deste trabalho. Foram incluídos trabalhos publicados em periódicos, conferências ou livros com revisão por pares, que abordassem explicitamente estratégias de economia de energia em RSSFs, apresentassem modelos/protocolos/algoritmos aplicáveis a nós alimentados por bateria e trouxessem resultados quantitativos ou experimentais sobre consumo, vida útil de rede ou desempenho; foram excluídos artigos sem relação direta com consumo energético (por exemplo, segurança, topologia ou mobilidade), duplicados entre bases, sem dados experimentais ou sem clareza metodológica e textos sem acesso ao conteúdo completo.

As bases de dados utilizadas nas consultas dos trabalhos correlatos foram a *IEEE Xplore, ScienceDirect, ACM Digital Library* e *Google Scholar*. O levantamento considerou publicações entre 2000 e 2024. O marco inicial (2000) foi escolhido por coincidir com a publicação do protocolo *LEACH*, referência seminal em gerenciamento energético em RSSFs, e o limite superior corresponde ao período de desenvolvimento deste trabalho. Foram utilizados descritores em português e inglês para garantir maior abrangência dos resultados, incluindo: “*energy management*”, “*wireless sensor networks*”, “*energy-efficient routing*”, “*energy consumption*”, “*duty cycling*”, “MAC *protocol*”, “gerenciamento de energia”, “redes de sensores sem fio”, “roteamento energético eficiente” e “protocolo de baixo consumo”.

Após a coleta inicial, os resultados foram organizados e filtrados manualmente. A leitura dos resumos e conclusões permitiu identificar os estudos mais relevantes, que foram analisados integralmente. Em seguida, os trabalhos selecionados foram categorizados conforme o tipo de abordagem: roteamento

hierárquico (LEACH), controle de acesso ao meio (B-MAC), classificações comparativas de estratégias energéticas, e gerenciamento dinâmico baseado na capacidade da bateria (Sousa *et al.*, 2007). Essa categorização orientou a estrutura de apresentação dos trabalhos correlatos descritos a seguir.

3.2 Revisão dos trabalhos correlatos

O gerenciamento de energia em RSSF é um tema amplamente discutido na literatura, em razão das severas restrições energéticas dos nós sensores e da necessidade de garantir autonomia e confiabilidade no funcionamento da rede. Diversas abordagens têm sido propostas, com destaque para o uso de roteamento baseado em energia residual, *duty cycling*, protocolos de comunicação de baixo consumo, escalonamento de tarefas, além da integração com fontes de energia renovável.

Um dos trabalhos pioneiros na área de gerenciamento energético em RSSF é o de Heinzelman *et al.* (2000), que propuseram o protocolo LEACH *(Low-Energy Adaptive Clustering Hierarchy*). Este protocolo introduziu uma abordagem de roteamento hierárquico adaptativo, onde os nós sensores são organizados em *clusters*, e cada cluster possui um nó líder (*cluster head*) responsável por receber os dados dos demais nós do grupo, agregá-los e então enviá-los à estação base. A grande sacada do LEACH é que esse papel de líder é rotativo entre os nós, com o objetivo de distribuir o consumo de energia ao longo do tempo e evitar que um único nó se esgote rapidamente. O LEACH também utiliza uma estratégia local de decisão, em que cada nó decide de forma probabilística se será ou não o líder em uma rodada específica, com base na energia residual e na frequência com que já foi líder anteriormente. Isso reduz a dependência de uma coordenação central e permite operações mais distribuídas e escaláveis, características fundamentais em ambientes com centenas ou milhares de sensores. Embora os resultados de simulação demonstram ganhos expressivos na vida útil da rede, com até 40% de redução no consumo energético total comparado a abordagens planas, sua aplicação prática enfrenta alguns desafios. Entre eles estão: a necessidade de alta sincronização temporal entre os nós, para garantir o funcionamento coordenado dos clusters, o aumento do overhead computacional nos nós líderes, que realizam funções extras de agregação e transmissão e a fragilidade frente a falhas de cluster heads, que podem comprometer grandes porções da rede. Apesar dessas

limitações, o LEACH serviu como base para o desenvolvimento de inúmeros protocolos derivados (como o LEACH-C, TEEN, APTEEN e LEACH-M), consolidando-se como um marco teórico e prático para estudos que visam o balanceamento energético em RSSFs.

Já Polastre *et al.* (2004) desenvolveram o protocolo B-MAC (*Berkeley Media Access Control*), uma solução voltada à camada de enlace (MAC) com o objetivo de minimizar o consumo energético em redes de sensores sem fio por meio de mecanismos de duty cycling adaptativo e escuta preditiva (low power listening). O B-MAC tem uma estrutura simples, modular e eficiente, que permite o desligamento periódico do rádio dos nós sensores para evitar o consumo desnecessário de energia durante períodos de ociosidade. O funcionamento do protocolo se baseia na transmissão de preâmbulos longos, que garantem que os receptores, mesmo em estado de sono, possam ligar a tempo de receber os dados. Esse modelo de escuta preditiva reduz o tempo em que o rádio permanece ativo, gerando uma economia de energia considerável sem comprometer a latência ou a confiabilidade da comunicação, especialmente em aplicações de baixa taxa de transmissão.

O B-MAC foi implementado e utilizado no sistema operacional TinyOS, um dos mais populares ambientes embarcados voltados para RSSFs, e passou por testes reais em redes experimentais, o que validou sua viabilidade prática. Resultados experimentais indicaram que o B-MAC consome até cinco vezes menos energia do que protocolos predecessores como o S-MAC, especialmente em cenários com baixa densidade de tráfego. Apesar de sua eficiência, o B-MAC também possui limitações. Seu uso de preâmbulos longos pode introduzir overhead em redes com alta densidade ou tráfego elevado, e a ausência de um mecanismo mais refinado de sincronização pode afetar o desempenho em aplicações com requisitos de tempo real. Ainda assim, o B-MAC é amplamente reconhecido como uma das primeiras implementações MAC verdadeiramente energeticamente eficientes, influenciando diretamente protocolos subsequentes.

Outro trabalho é o estudo realizado por Vidhyapriya e Vanathi (2007), que apresenta uma classificação e uma análise comparativa de estratégias de roteamento energeticamente eficientes aplicadas à RSSF. Os autores propõem uma taxonomia funcional dos protocolos de roteamento, dividindo-os em três grandes categorias: proativos, reativos e híbridos. A abordagem visa compreender como diferentes técnicas podem impactar o consumo de energia e a eficiência global da

rede. Um dos principais pontos discutidos pelos autores é a necessidade de adaptação das rotas com base na energia residual dos nós sensores, o que permite prolongar a vida útil da rede e prevenir falhas em cascata decorrentes da falta de energia em determinados nós. Além disso, o trabalho destaca a importância de estratégias descentralizadas, nas quais os próprios nós sensores tomam decisões localmente sobre participação em rotas ou atuação, com base em parâmetros internos como nível de energia, conectividade ou proximidade com a estação base.

O trabalho que mais se assemelha ao presente trabalho foi realizado por Sousa *et al*. (2007). Eles apresentaram uma abordagem para o gerenciamento dinâmico de energia em redes de sensores sem fio, baseada em um formalismo matemático denominado Redes de Petri Híbridas Diferenciais (RPHDs). Este formalismo permite representar, de forma integrada, tanto a dinâmica a eventos discretos, como a ativação de sensores e transmissões, quanto a dinâmica contínua, como a descarga e recuperação da bateria, características típicas de sistemas híbridos como as RSSFs. A proposta dos autores é chamada de GDC (Gerenciamento Dinâmico baseado na Capacidade da bateria). Ela consiste em desligar os subsistemas de comunicação e sensoriamento dos nós sensores após determinadas transmissões, dependendo da capacidade restante da bateria. O desligamento ocorre por breves períodos (por exemplo, 2 ms), permitindo que a bateria se recupere parcialmente — um fenômeno real conhecido como efeito de recuperação. Essa técnica busca distribuir os momentos de repouso ao longo do ciclo de operação do nó sensor, em vez de realizar longos períodos de desligamento contínuo, o que pode comprometer o funcionamento da rede.

Três modos de gerenciamento foram definidos:

- **GDC1:** ativado entre 65% e 80% da capacidade da bateria, com desligamento após 6 transmissões;
- **GDC2:** entre 50% e 65%, com desligamento após 3 transmissões;
- **GDC3:** abaixo de 50%, com desligamento após cada transmissão.

Em simulações com mais de 500.000 operações, os autores observaram ganhos significativos:

- **GDC1**: 1,15% de recuperação da capacidade da bateria;
- **GDC2**: 2,34%;
- **GDC3**: até 5,60% — o modo mais eficiente.

A Figura 6 a seguir mostra de forma mais ampla a tabela de resultados do trabalho do Sousa *et al*. (2007), é possível notar o ganho de eficiência dos modos.

Figura 6 - Resultados do trabalho Sousa *et al*. (2007)

Modos de Operação	$C_{inicial}$ (mA-ms)	C_{final} (mA-ms)	Ganho (%)
Sem GDC	1967280,00	1761035,42	
GDC1 (Sleep = 2 ms)	1967280,00	1781332,83	1,15
Sem GDC	1598415,00	1392170,42	
GDC2 (Sleep = 2 ms)	1598415,00	1424758,86	2,34
Sem GDC	1229550,00	1023305,42	
GDC3 (Sleep = 2 ms)	1229550,00	1080625,53	5,60

Fonte: Sousa *et al.* (2007)

Esses resultados foram obtidos com o uso do modelo de bateria Rakhmatov-Vrudhula, que considera o comportamento não linear e o efeito de recuperação, tornando a estimativa de consumo energético mais realista.

A abordagem de Sousa *et al.* pode ser aplicada em cenários em que os nós sensores são alimentados apenas por baterias e instalados em locais de difícil acesso. Embora a proposta utilize modelagem formal, o princípio é altamente aplicável a sistemas embarcados reais como o deste trabalho. Assim como na proposta atual, o objetivo é estender a vida útil do nó sensor por meio do controle eficiente de energia com base no estado da bateria.

Desta forma, a revisão dos trabalhos correlatos evidenciou avanços como LEACH e B-MAC e apontou lacunas de simplicidade e aplicabilidade em cenários reais de baixo custo. O estudo de Sousa et al. (2007) reforça a eficácia de políticas dinâmicas baseadas no estado da bateria, base conceitual da abordagem aqui proposta. No capítulo seguinte, descreve-se a metodologia adotada para projetar e validar esse controle adaptativo de energia no nó sensor.

4. Metodologia

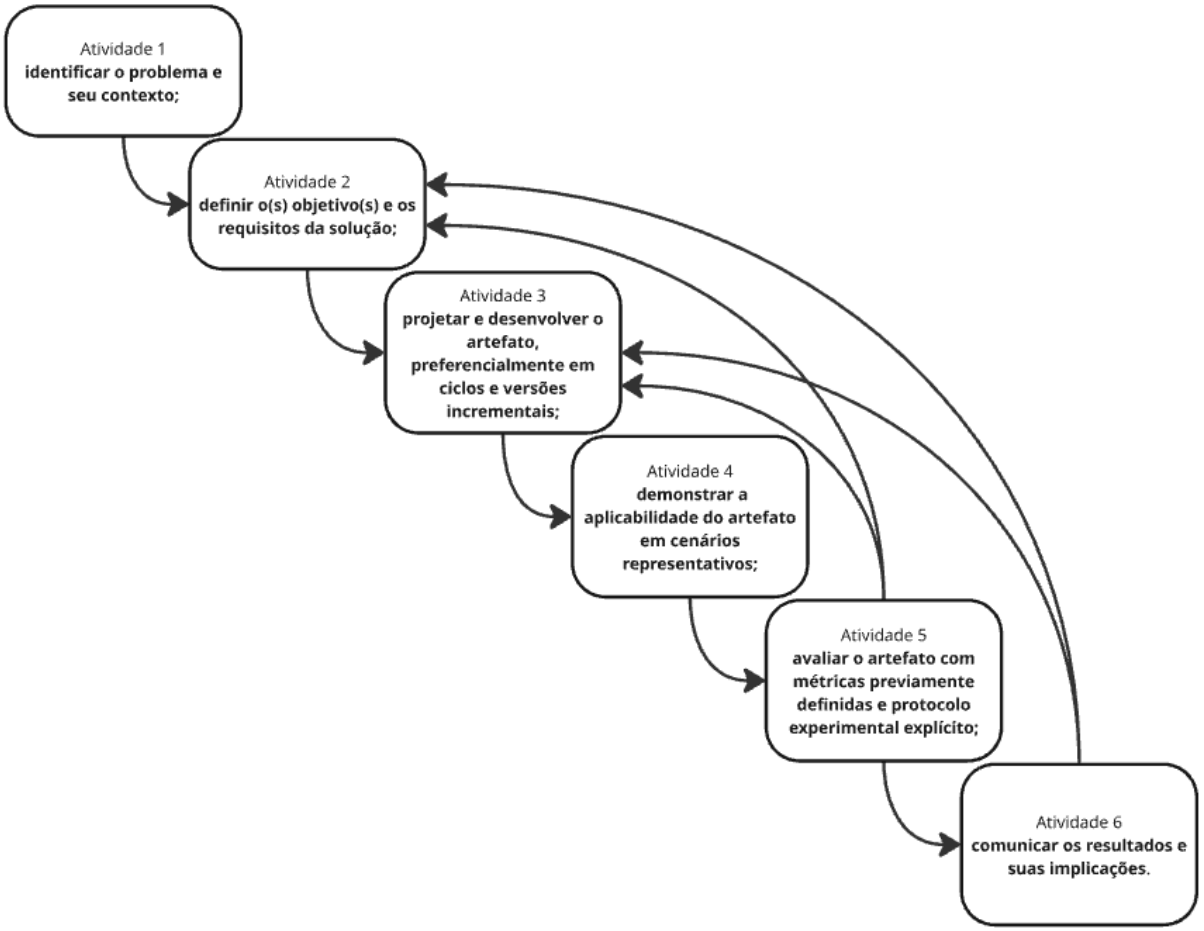
4.1 Tipo de Pesquisa

A presente pesquisa é aplicada, com abordagem quantitativa e de caráter experimental, direcionada à projeção, construção e avaliação de um artefato tecnológico concebido para gerenciamento inteligente de energia em nó sensor, combinando algoritmo embarcado e interface web para supervisão e intervenção do usuário. O foco central é elevar a autonomia da bateria sem comprometer a disponibilidade de serviços essenciais, equilibrando consumo energético, desempenho das funcionalidades e segurança operacional do sistema. Em termos metodológicos, o estudo envolve definição explícita de requisitos, implementação incremental do artefato, e ensaios controlados que permitem quantificar efeitos sobre métricas de interesse, por exemplo, autonomia em horas, consumo médio em mA e disponibilidade dos serviços essenciais.

Nesse contexto, a *Design Science Research* (DSR) se mostra particularmente adequada porque orienta pesquisas cujo objetivo é criar e aperfeiçoar artefatos para resolver problemas práticos, garantindo rigor por meio de ciclos iterativos de projeto. A DSR não apenas legitima a intervenção tecnológica como parte do método científico, mas também exige que o artefato seja avaliado segundo critérios objetivos e replicáveis, alinhando-se ao desenho deste trabalho: há um problema real, restrição energética em RSSF, uma solução implementável, uma abordagem de gerenciamento energético com uma política automática que altera os modos de funcionamento do nó sensor através de um algoritmo embarcado implementado no microcontrolador que faz a leitura da tensão das baterias e faz a transição entre os modos com limiares e histerese, mais a priorização de serviços essenciais via interface web, e um protocolo experimental que compara condições para mensurar o ganho de desempenho.

Em que consiste a DSR (síntese): Em termos gerais, a *Design Science Research* organiza-se em seis atividades, ver Figura 7: **(i)** identificar o problema e seu contexto; **(ii)** definir o(s) objetivo(s) e os requisitos da solução; **(iii)** projetar e desenvolver o artefato, preferencialmente em ciclos e versões incrementais; **(iv)** demonstrar a aplicabilidade do artefato em cenários representativos; **(v)** avaliar o artefato com métricas previamente definidas e protocolo experimental explícito; e **(vi)** comunicar os resultados e suas implicações.

Figura 7. Processo metodológico para DSR.



Fonte: Adaptado de Peffers *et al.* (2007).

4.2 Etapas de pesquisa da DSR e o que foi realizado no trabalho

Seguindo as etapas de pesquisa da DSR, no primeiro capítulo deste trabalho foram identificados o contexto e o problema, a limitação de autonomia em nós de RSSF alimentados por bateria e a ausência de uma política adaptativa sob restrição energética. Em seguida, foi realizada uma revisão de trabalhos correlatos para compreender abordagens existentes e lacunas. A partir desse diagnóstico, foi formulada uma hipótese de solução e definidos os objetivos da pesquisa, que orientam o desenvolvimento do artefato proposto. Para entender melhor o problema

e ajudar no desenvolvimento do artefato da solução, construiu-se o referencial teórico para nortear a nossa pesquisa. Na sequência, o projeto avançou para o desenvolvimento do artefato, materializando a solução em um sistema funcional que pode gerenciar o consumo energético de um nó sensor de maneira automática com um algoritmo embarcado ou de maneira manual via interface web que se comunica com o nó. Com o artefato implementado, procedeu à sua demonstração em cenários controlados e à avaliação dos resultados, gerando evidências para discussão e comunicação dos achados conforme o encadeamento proposto pela DSR.

4.3 Método de Desenvolvimento do artefato

Esta seção descreve como o artefato foi projetado e opera, detalhando a plataforma do nó, a lógica de estados baseada na tensão da bateria (Vbat), os limiares com histerese e o protocolo de transição entre modos, informações necessárias para avaliação dos resultados.

Para desenvolver o artefato proposto neste trabalho, primeiro foi preciso construir um nó sensor, que serviu para testar o artefato e validar a pesquisa. A plataforma do nó sensor conta com um Arduino Mega que atuou como microcontrolador dos sensores e um ESP8266 para coordenar e se comunicar com o servidor que recebeu e tratou os dados do nó. A plataforma é alimentada por um módulo de energia, aqui utilizamos uma fonte de bancada DC ajustada em 12V para simular o uso de baterias de lítio recarregáveis que seria a tensão ideal para esse nó sensor com essas especificações. Para a leitura de tensão da bateria (Vbat) feita pelo Arduino Mega integramos um sensor de tensão calibrado na entrada de energia do nó sensor. A comunicação do ESP8266 com o Arduino ocorre via Serial.

Com o nó sensor criado, foi definido que o gerenciamento de energia do nó iria ser feito de acordo com a Vbat, de acordo com o datasheet do Arduino Mega, a sua tensão de operação nominal está entre 7V a 12V, com isso definimos no nosso projeto que 7V no nosso módulo de energia representa 5% de bateria do nó e 12V representa 100% de bateria. Com isso foi definido os modos de operação do nó sensor, sendo eles: NORMAL, ECONOMIA, CRÍTICO e RESERVA. A leitura da tensão é convertida em porcentagem no algoritmo embarcado do Arduino Mega. Para obter maior precisão na definição dos modos, utiliza-se histerese a fim de evitar trocas frequentes entre eles. Cada modo determina prioridades e estabelece quais

portas e sensores permanecem ligados. O estado pode ser forçado manualmente pela interface web ou selecionado automaticamente pelo firmware conforme as faixas de Vbat. No Quadro 1 apresentam-se as faixas de tensão, emulando um sensor alimentado por bateria de 12V, a função de cada modo e as ações típicas.

Quadro 1 – Estados de operação, faixas de tensão (emulação 12 V) e função

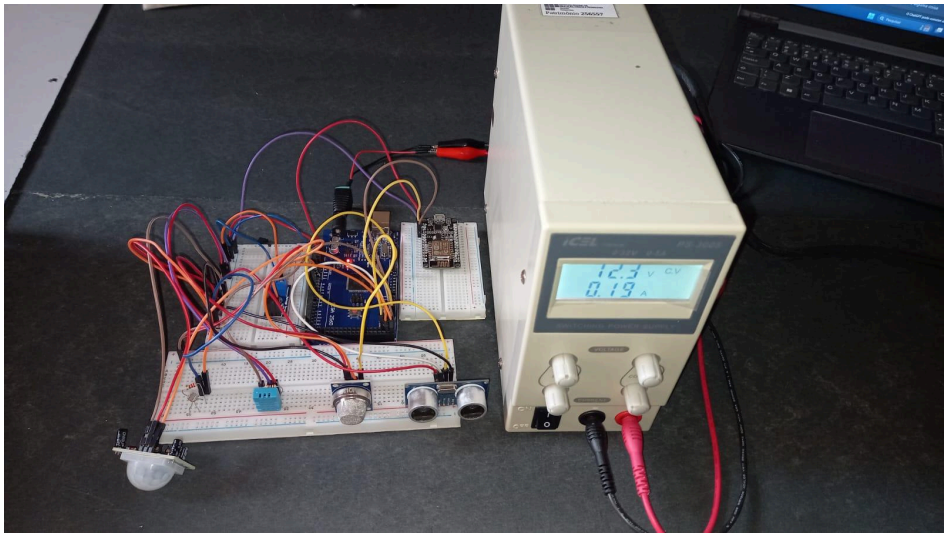
Modo de operação	Faixa de tensão / Porcentagem de bateria	Função
NORMAL	Vbat ≥ ~9,37 V | ≥50%	Operação nominal
ECONOMIA	~9,32 – 8,05 V | 49–25%	redução de taxa de amostragem e desligamento de módulos definidos na montagem do nó
CRÍTICO	~8,00–7,26 V | 24–10%	manter apenas serviços essenciais definidos na montagem do nó
RESERVA	Vbat < ~7,26 V | <10%	Todos os sensores desligados. Objetivo: aguardar recarga externa do módulo de energia renovável solar ou eólico até que a bateria atinja limiar seguro para retornar aos demais modos

O algoritmo desenvolvido tem foco em priorizar a autonomia do nó sensor, ele reduz sensores e frequência de leitura conforme a bateria cai. O programa inicia definindo o modo atual como NORMAL e zera o marcador de última leitura. Em cada iteração do loop principal, ele mede a tensão da bateria, converte essa tensão em porcentagem de carga e, a partir disso, decide qual deve ser o próximo modo de operação (NORMAL, ECONOMIA, CRÍTICO ou RESERVA). Se o modo decidido for diferente do modo atual, o sistema atualiza o modo e aplica a política correspondente, que basicamente liga/desliga trilhos de energia dos periféricos e define o intervalo de amostragem apropriado para aquele modo. Com o modo já estabelecido, o programa consulta a política ativa para saber qual é o intervalo de registro. Só quando o tempo desde o último registro atinge esse intervalo é que ele atualiza e emite um log contendo a tensão e a porcentagem de bateria, o nome do modo atual e o estado das linhas de potência dedicadas a cada periférico. Esse ciclo

se repete indefinidamente, ajustando o modo conforme a bateria cai ou se recupera e respeitando o ritmo de registro definido pela política do modo. Além disso, os modos também podem ser definidos manualmente através da interface web.

Para realizar os testes, está sendo usado uma fonte de bancada DC (Corrente direta) programável no lugar da bateria física, emulando uma bateria de lítio de 12V recarregável, cenário que simula um sensor com uma fonte de energia renovável solar ou eólica. São aplicados perfis: (a) rampa de descarga, simula o uso do nó sensor, (b) degraus para testar a histerese e (c) rampa de recarga, simula a bateria do sensor sendo alimentado pelo módulo de energia renovável, este teste serve para validar a saída do modo RESERVA do nó sensor. A Figura 9 apresenta o protótipo conectado à fonte de bancada.

Figura 9 - Figura 9 – Protótipo em operação com alimentação fornecida por fonte de bancada.



Fonte: Do autor.

A histerese estabelece dois limiares para cada transição de modo: um limiar de entrada, acionado na queda da tensão da bateria, e um limiar de retorno, ligeiramente mais alto que é acionado na subida. Essa faixa de segurança evita comutações frequentes próximas aos limites, estabiliza o comportamento do nó e reduz liga/desliga desnecessário de sensores, rádio e do próprio microcontrolador. Na prática, o firmware só desce de um modo para outro quando Vbat fica abaixo do limiar de entrada e só retorna quando Vbat supera o limiar de retorno. Para a emulação com alimentação de 12V do nó, adotou-se uma largura de retorno de

histerese (ΔV_ret) típica de 0,20–0,30V, valor suficiente para superar ruído de medição, variações momentâneas de carga e quedas transitórias por picos de corrente. No Quadro 2 apresenta-se a parametrização utilizada nos ensaios de bancada, com os limiares de entrada e retorno para cada transição, bem como a finalidade de cada escolha.

Quadro 2 – Parâmetros de histerese (limiares de entrada e retorno) para o nó sensor a 12 V

Transição	Entrada (queda de tensão)	Retorno (subida de tensão)	ΔV_ret	Observação
NORMAL -> ECONOMIA	9,37V	9,62V	0,25V	Evita oscilar entre operação nominal e economia por pequenas variações próximas a 9,4 V.
ECONOMIA -> CRÍTICO	8V	8,30V	0,30V	Garante que só volta à Economia após recuperação consistente.
CRÍTICO -> RESERVA	7,26V	7,46V	0,20V	Em Reserva, o nó dorme e acorda periodicamente; retorno exige tensão um pouco acima do limiar.

Elaborado pelo autor.

Lógica de implementação:

1. Transição por entrada: quando Vbat cai abaixo do limiar de entrada do próximo modo, o sistema altera o modo.
2. Retorno com histerese: o sistema só retorna ao modo anterior quando Vbat ultrapassa o limiar de retorno, definido como *(limiar de entrada + ΔV_ret)*.

3. Comportamento em RESERVA: no modo RESERVA, o nó entra em sono profundo (*deep sleep*) e acorda periodicamente apenas para medir Vbat; só sai desse modo quando atender aos critérios de retorno.

4.4 Procedimentos de teste

Este capítulo descreve como o artefato será testado e avaliado, em condições de bancada que emulam uma bateria de 12 V sob descarga e recarga controladas. O objetivo é verificar, de forma reprodutível, se o nó sensor se comporta conforme o esperado nos dois modos de operação do usuário: (i) controle manual via interface web (seleção explícita dos modos Normal, Economia, Crítico e Reserva) e (ii) controle automático (troca de modos guiada pela tensão/porcentagem da bateria com histerese). A seguir, apresenta-se o passo a passo dos ensaios correspondentes aos cenários (i) e (ii), detalhando as etapas de preparação, execução e registro para cada caso.

- **Teste (i)** — Controle manual via interface web:

Objetivo: quantificar quanto cada modo consome com a sua configuração real de portas/sensores.

1. Alimentação: ligar o nó sensor em 12V na fonte de bancada.
2. Portas/Sensores: definir quais portas ficam ligadas em cada modo.
3. Configuração na UI: na interface web, selecionar manualmente entre os modos (Normal, Economia, Crítico ou Reserva).
4. Medição: medir a corrente diretamente na fonte de bancada e com multímetro e registrar o valor para cada modo após estabilização.
5. Repetição: repetir as medições pelo menos 3 vezes por modo para obter média e variação.

- **Teste (ii)** — Troca automática de modo pelas faixas de porcentagem/tensão

Objetivo: verificar se o algoritmo automático entra e sai dos modos sem intervenção, conforme as faixas de tensão/porcentagem e a histerese.

1. Descarga simulada: com o nó ligado, reduzir gradualmente a tensão na fonte (de 12V para baixo), simulando descarga.
2. Observação das transições: confirmar que o firmware altera os modos sozinho conforme as faixas (Normal -> Economia -> Crítico -> Reserva) e que não oscila perto dos limites, histerese funcionando.
3. Recarga simulada: aumentar a tensão para simular recarga e confirmar o retorno (Reserva -> Crítico/Economia -> Normal) respeitando os limiares de retorno.
4. Repetição: repetir o ciclo descarga/recarga ao menos 2–3 vezes para consistência.

5. Artefato: projeto, implementação e modelagem

Este capítulo descreve em detalhe o desenvolvimento do trabalho e a construção do artefato proposto, que integra soluções de *hardware* e *software* voltadas ao gerenciamento do consumo energético de um nó sensor.

5.1 Hardware

Começando pela camada de *hardware*, foi necessária a construção de um nó sensor dedicado à implementação e validação experimental do artefato proposto. O protótipo desenvolvido é composto por cinco sensores distintos (ver Quadro 3), empregados de forma representativa, com o propósito de emular diferentes demandas energéticas e fluxos de dados.

Quadro 3 - Sensores do nó para simulação

Nome	Descrição
HC-SR04	Sensor de distância
PIR HC-SR501	Sensor de movimento
DHT11	Sensor de temperatura e umidade
LDR	Sensor de luminosidade
MQ-2	Sensor de fumaça

A função específica de cada sensor não constitui o foco central deste trabalho, uma vez que a arquitetura projetada é genérica e pode ser adaptada a diversos contextos de RSSF, mantendo sua aplicabilidade independente do tipo de grandeza monitorada. Assim, o conjunto sensorial atua como uma carga experimental para testar a eficiência do mecanismo de gerenciamento energético implementado.

Após a montagem do nó, foi integrado à entrada de alimentação um sensor de tensão, responsável por monitorar continuamente a Vbat. Essa leitura é processada pelo microcontrolador, permitindo estimar o percentual de carga da bateria e, com base nela, realizar a transição entre os modos de operação definidos no algoritmo embarcado. Esse controle é fundamental para o funcionamento do sistema de gerenciamento energético, pois possibilita que o nó alterne entre estados de consumo reduzido e normal conforme a disponibilidade energética, assegurando maior autonomia e estabilidade do dispositivo. Além disso, o nó sensor também

incorpora um módulo ESP8266, que faz o papel de interface de comunicação entre o nó e o servidor. Ele atua como uma ponte bidirecional: de um lado, envia periodicamente dados de telemetria e estado energético; de outro, recebe comandos oriundos da interface web, permitindo ao usuário forçar manualmente a troca de modo de operação, além de monitorar em tempo real o nível de carga e o status dos sensores. Essa integração viabiliza tanto o gerenciamento automático, conduzido pelo algoritmo embarcado, quanto o gerenciamento manual, por meio da interface de supervisão. As ligações de hardware, incluindo o esquema de conexões, portas utilizadas e configuração dos sensores, encontram-se detalhadas no Apêndice C.

5.2 *Software*

Para a camada de *software*, foi desenvolvido um algoritmo embarcado em linguagem C do Arduino (vide código disponível no Apêndice A). Este código é responsável pelo funcionamento dos sensores do nó, realizando as leituras pelas quais cada um é responsável, e também faz toda a parte de gerenciamento de energia através do nível de bateria do nó ou através da troca manual de modos de operação. Destrinchando o código, primeiro é inicializado todas as variáveis importantes, como os modos de operação, depois definimos a política de funcionamento de cada método, ou seja, quais portas e quanto tempo de leitura cada modo de operação deve respeitar.

O nó inicia sua operação em Modo Normal, com a contagem de tempo zerada e o controle manual desativado, garantindo que o comportamento inicial seja totalmente automático. A partir desse ponto, ele entra em um ciclo contínuo de execução, que se repete indefinidamente enquanto o sistema estiver energizado. Em cada iteração desse ciclo, o nó realiza a leitura da bateria por meio de um sensor de tensão conectado à entrada analógica. Essa leitura é feita diversas vezes e a média das amostras garante estabilidade e evita ruídos elétricos. O valor analógico obtido é então convertido em tensão elétrica (volts) considerando a referência de 5 V e o fator de divisão do circuito. Com a tensão real medida, o sistema executa uma conversão proporcional para porcentagem de carga: adota-se uma escala em que 7 volts representam aproximadamente 5% da capacidade (nível mínimo operacional) e 12 volts equivalem a 100% da carga total. Entre esses limites, é feita uma interpolação linear, permitindo estimar com precisão o nível de bateria em qualquer ponto

intermediário. Após determinar essa porcentagem, o nó a utiliza como base para tomar decisões de energia. Assim, a cada ciclo o sistema verifica se deve manter o modo atual ou mudar para outro — como Economia, Crítico ou Reserva — de acordo com a condição da bateria. Esse processo contínuo garante que o nó adapte automaticamente seu consumo, preservando a autonomia e assegurando que os sensores e módulos permaneçam ativos apenas quando há energia suficiente para operá-los com segurança.

Além dessa mudança de modo de maneira automática pela leitura Vbat, também é possível a troca de modos de maneira atual, assim, se houver um comando vindo da interface web escolhendo um modo específico, o equipamento prioriza essa ordem. A partir desse instante, o controle manual entra em ação: a lógica automática fica suspensa, e o nó permanece fixo no modo indicado pelo usuário, independentemente das novas leituras de bateria. Esse estado é sinalizado nos registros e mantido até que ocorra uma dessas condições: (a) o usuário envie um novo comando manual trocando o modo; ou (b) o controle manual seja desativado, momento em que a decisão automática é restabelecida e volta a considerar a porcentagem de bateria com os limiares e histerese. Enquanto o controle manual estiver ativo, não há mudança de modo induzida por variações de tensão; apenas ações explícitas do usuário alteram o estado operacional.

Sempre que a escolha (manual ou automática) muda de modo, o sistema aplica a “política” daquele modo. Essa política define, por exemplo, o que fica ligado ou desligado e de quanto em quanto tempo o equipamento deve registrar informações. Em outras palavras: cada modo ajusta quais recursos ficam ativos e com que frequência o sistema trabalha, para equilibrar o consumo de energia com a necessidade de dados. Depois de aplicar a política do modo atual, o equipamento espera até que chegue o próximo momento de registro, de acordo com o intervalo definido por esse modo. Quando esse momento chegar, ele grava um resumo: a tensão e a porcentagem da bateria, o nome do modo em uso (e uma marca de “forçado” caso esteja sob controle manual) e o estado das linhas de energia dos componentes. Em seguida, o ciclo recomeça, repetindo o processo: mede a bateria, verifica se há comando externo, decide o modo apropriado, aplica a política e registra no tempo certo.

5.2.1 Servidor e interface Web

O servidor responsável pela comunicação e armazenamento de dados foi desenvolvido em PHP, utilizando o MySQL como sistema gerenciador de banco de dados, administrado por meio do phpMyAdmin. O servidor atua como o núcleo central de processamento, recebendo periodicamente as leituras enviadas pelo nó sensor, como nível de bateria, temperatura e demais variáveis, e armazenando-as em tabelas estruturadas. Essas informações podem ser posteriormente consultadas pela interface web, permitindo o monitoramento em tempo real do estado da rede. Além disso, o servidor também é responsável por enviar comandos de controle ao nó sensor, possibilitando a troca manual dos modos de operação e o ajuste de parâmetros conforme necessidade do usuário.

A interface web foi desenvolvida utilizando HTML, CSS e JavaScript, tecnologias que possibilitam a criação de um ambiente leve, responsivo e acessível a partir de qualquer navegador moderno. Essa interface tem como principal função permitir ao usuário acompanhar os dados coletados e interagir com o nó sensor, enviando comandos de forma simples e intuitiva.

Todo o sistema foi hospedado na nuvem da Hostinger, o que garante maior disponibilidade, escalabilidade e acesso remoto à aplicação. Dessa forma, o servidor e a interface web atuam conjuntamente como a camada de supervisão do sistema, possibilitando tanto o gerenciamento automático, conduzido pelo algoritmo embarcado, quanto o gerenciamento manual, realizado pelo usuário por meio do painel online. A escolha dessa arquitetura se deu pelo conhecimento prévio nessas tecnologias, o que permitiu maior agilidade na implementação, manutenção e integração entre os módulos do sistema.

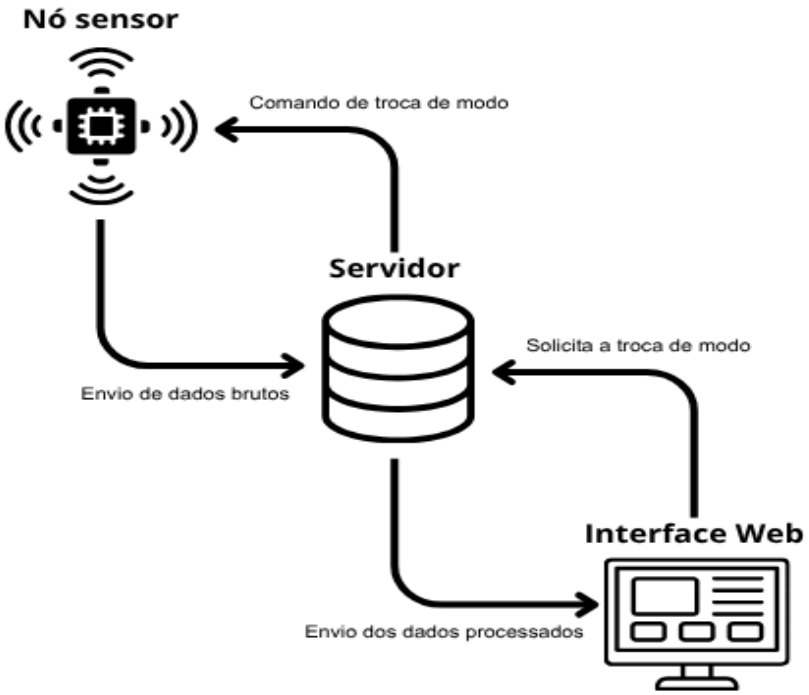
Figura 8 - Interface Web

Fonte: Do autor

5.2 Arquitetura geral

A arquitetura geral do sistema foi concebida de forma modular, integrando as camadas de hardware, software embarcado e infraestrutura web em um fluxo de comunicação contínuo. O nó sensor realiza as medições ambientais e o monitoramento do nível de bateria, enviando periodicamente esses dados ao servidor desenvolvido em PHP e MySQL. O servidor, por sua vez, processa, armazena e disponibiliza as informações para a interface web, onde o usuário pode acompanhar os dados em tempo real e, se desejar, enviar comandos manuais de controle ao nó. Essa estrutura garante a interação bidirecional entre o dispositivo físico e a aplicação online, permitindo tanto o gerenciamento automático de energia quanto o controle remoto pelo usuário. A Figura 9 a seguir apresenta uma visão geral da arquitetura do sistema, destacando o fluxo de dados entre as camadas.

Figura 9 - Arquitetura geral do sistema



Fonte: Do autor

6. Resultado e Discussões

Este capítulo apresenta os resultados obtidos a partir da simulação e análise de consumo energético do nó sensor desenvolvido, considerando as quatro faixas de operação do algoritmo de gerenciamento inteligente de energia: Normal, Economia, Crítico e Reserva. O nó foi composto pelos sensores DHT11, LDR, PIR

HC-SR501, HC-SR04, MQ-2, além do módulo de tensão, do Arduino Mega 2560 e do ESP8266, responsável pela comunicação com o servidor. Devido à limitação de tempo durante a execução do projeto, não foi possível realizar medições contínuas com sensor de corrente dedicado, o que seria o cenário ideal para testes laboratoriais mais precisos.

Dessa forma, as estimativas apresentadas baseiam-se em valores médios de consumo extraídos de datasheets e em medições pontuais com multímetro digital. Essa metodologia, embora simplificada, fornece resultados coerentes com o comportamento esperado de sistemas embarcados equivalentes e é adequada para a validação conceitual do artefato.

6.1 Prioridade dos Sensores

Durante o desenvolvimento do nó sensor, foi necessário definir níveis de prioridade entre os sensores, determinando quais permaneceram ativos em cada faixa de operação. Essa priorização segue o princípio de manter sensores essenciais ligados por mais tempo, assegurando que as funções mais críticas continuem disponíveis mesmo sob baixa energia.

O Quadro 4 apresenta a relação de sensores ativos em cada modo de operação:

Quadro 4 – Sensores ativos conforme a faixa de operação (1 ligado, 0 desligado)

Faixa	**DHT11**	**LDR**	**PIR**	**HC-SR 04**	**MQ-2**	**Intervalo de leitura**	**Modo ESP**
Normal (≥50%)	1	1	1	1	1	2s	Sempre ativo
Economia (49–25%)	1	1	1	0	0	3,5s	modem sleep
Crítico (24–10%)	1	1	1	0	0	5s	modem sleep
Reserva (<10%)	1	1	0	0	0	30s (apenas leitura da bateria)	deep sleep

Com essa configuração, o modo Reserva mantém apenas sensores essenciais de ambiente (DHT11 e LDR), garantindo que o nó continue reportando temperatura, umidade e luminosidade mesmo em níveis críticos de energia. Essa priorização é comparável a dispositivos móveis, que restringem funções não essenciais ao entrar em modo de economia de bateria.

6.2 Consumo energético estimado

Os valores de corrente total foram calculados somando o consumo de todos os módulos ativos em cada faixa, considerando o ESP8266 com diferentes níveis de operação — ativo no modo Normal, *modem-sleep* nos modos intermediários e *deep-sleep* no modo Reserva. A tensão nominal utilizada nos cálculos foi de 12V, e a bateria simulada possui 12V / 7Ah, equivalente a 84 Wh.

Quadro 5 – Consumo energético e autonomia por faixa

Faixa	**Corrente (A)**	**Potência (W)**	**Consumo/dia (Wh)**	**Autonomia (h)**
Normal	0,30	3,60	86,40	23,3
Economia	0,17	2,04	48,96	41,2
Crítico	0,13	1,56	37,44	53,8
Reserva	0,083	0,99	23,88	84,4

Observa-se que a redução gradual de sensores e o ajuste de potência do ESP8266 resultam em uma economia superior a 70% no consumo diário entre o modo Normal e o modo Reserva. Essa diferença demonstra a eficiência do gerenciamento adaptativo proposto.

6.3 Comparativo de autonomia e comportamento adaptativo

Para avaliar o impacto prático do gerenciamento energético proposto, foram simuladas as condições de operação do nó sensor desenvolvido neste trabalho em diferentes faixas de carga da bateria. As análises consideram uma bateria de 12V / 7Ah (≈84 Wh) e o consumo energético medido ou estimado para cada modo de operação, Normal, Economia, Crítico e Reserva, conforme as combinações de sensores e estados do módulo de comunicação definidos no projeto.

No modo Normal, o nó opera com todos os sensores e o Wi-Fi em funcionamento contínuo, resultando em um consumo aproximado de 3,6 W. Já nas faixas subsequentes, o sistema passa a desligar sensores de maior demanda e

reduzir a frequência de leitura, além de modificar o modo de operação do ESP8266 para *modem-sleep*, *light-sleep* ou *deep-sleep*, diminuindo o consumo progressivamente até alcançar 0,99 W no modo Reserva — em que permanecem apenas os sensores DHT11 e LDR, considerados essenciais no nó sensor deste trabalho.

Essa redução gradual no consumo gera reflexos diretos na autonomia. O modo Normal, que representa a condição sem gerenciamento adaptativo, alcança cerca de 23,3 horas de autonomia contínua. Com a aplicação do algoritmo, a faixa de Economia eleva a autonomia para 41,2 horas, e a faixa Crítica chega a 53,8 horas. No modo Reserva, o nó atinge 84,4 horas de funcionamento, mesmo com sensores mínimos ativos, representando um ganho total de mais de 260% em relação ao modo fixo.

Para demonstrar o efeito do gerenciamento nos estágios finais de descarga, foi simulada a situação em que a bateria atinge 10% da capacidade, equivalente a 8,4 Wh restantes:

1. No primeiro cenário desta simulação sem gerenciamento adaptativo (modo fixo Normal): O nó consome 3,6 W continuamente e esgota essa energia em aproximadamente 2,3 horas.
2. No segundo cenário da simulação com gerenciamento adaptativo (modo Reserva ativo): O consumo é reduzido para 0,99 W, estendendo o tempo de operação para 8,4 horas.

Essa diferença de mais de seis horas adicionais de funcionamento demonstra a eficiência da estratégia de transição automática entre faixas. Em contextos de campo, essa margem extra é fundamental, pois permite que a bateria seja recarregada antes do desligamento completo do nó, especialmente em sistemas alimentados por fontes renováveis, como painéis solares ou microgeradores eólicos. Assim, o nó mantém conectividade e capacidade mínima de coleta de dados durante o período de recarga, evitando lacunas nas medições e reduzindo a necessidade de intervenção manual.

Em síntese, os resultados obtidos evidenciam que o gerenciamento adaptativo não apenas prolonga a autonomia do nó, mas também aumenta a janela operacional para recarga da bateria, garantindo a continuidade dos serviços essenciais e tornando o sistema mais confiável para aplicações remotas e autônomas em redes de sensores sem fio.

6.4 Validação da mudança de modo via interface web

Outro aspecto testado foi a implementação da interface web para mudança manual dos modos de operação. Os ensaios mostraram que, ao acionar um modo na UI, o nó altera imediatamente o conjunto de sensores ativos, os intervalos de leitura e o estado do ESP8266 (ativo/light-sleep/deep-sleep), mantendo a prioridade do controle manual sobre o automático até sua liberação — exatamente como previsto nos procedimentos e objetivos do trabalho.

Figura 10 - Seleção de modo de operação na interface web

Fonte: do autor.

6.5 Discussão dos resultados

Os resultados demonstram que o gerenciamento adaptativo de energia aumentou de forma significativa a autonomia do nó sensor. A autonomia passou de 23,3 horas no modo convencional para 84,4 horas no modo adaptativo, representando um ganho de aproximadamente 262%. Além disso, o consumo energético diário foi reduzido em cerca de 72%, e, em condições críticas (10% de carga), o tempo de operação aumentou de 2,3 para 8,4 horas, um incremento

superior a 265%. Esses resultados confirmam a eficácia do modelo proposto em prolongar a vida útil da bateria e reduzir o risco de inatividade do sistema.

Assim como ocorre em smartphones que entram em modo de economia ao atingir níveis críticos de bateria, o nó sensor desenvolvido reduz automaticamente o consumo, mantendo apenas funções vitais para o monitoramento. Essa lógica garante que o sistema continue operando por um período suficiente até o restabelecimento da energia, aspecto essencial em aplicações autônomas e remotas.

Embora as medições tenham caráter estimativo, os resultados mostram coerência com os valores de consumo informados pelos fabricantes dos sensores e validam o modelo proposto. Recomenda-se, em trabalhos futuros, a utilização de sensor de corrente calibrado e testes de descarga reais com múltiplos ciclos, de modo a confirmar experimentalmente as curvas de consumo e ajustar com maior precisão os limiares de transição.

Em síntese, os resultados demonstram que o artefato cumpre seu objetivo principal de otimizar o consumo energético e aumentar a autonomia operacional, mostrando-se uma solução viável e adaptável para redes de sensores sem fio com restrição energética.

7. Conclusão

O desenvolvimento deste trabalho permitiu evidenciar que a aplicação de um gerenciamento adaptativo de energia é uma solução que possibilita aumentar a autonomia e a confiabilidade de nós sensores alimentados por bateria. Por meio da definição de faixas de operação, com diferentes combinações de sensores ativos e modos de funcionamento do módulo de comunicação, o sistema foi capaz de ajustar seu consumo conforme o nível de carga, mantendo a operação essencial mesmo em condições de energia reduzida.

Os resultados demonstraram que, com a estratégia proposta, o nó sensor apresentou um ganho expressivo de autonomia, alcançando até 84 horas de

funcionamento no modo de menor consumo, contra apenas 23 horas no modo convencional. Além disso, o comportamento de mudança automática e manual via interface web funcionou conforme o esperado, permitindo que o usuário controle remotamente os modos e validando o objetivo específico de integração entre hardware e interface. O modelo desenvolvido mostrou-se viável, simples de implementar e flexível, podendo ser adaptado para diferentes aplicações de monitoramento ambiental e sistemas de redes de sensores sem fio. O nó se comportou de maneira previsível, apresentando estabilidade tanto nas transições automáticas quanto nos comandos remotos, o que reforça sua aplicabilidade em cenários reais, especialmente em ambientes isolados com alimentação solar.

Como trabalhos futuros, recomenda-se:

1. A integração de inteligência artificial para prever padrões de consumo e ajustar dinamicamente as faixas de operação com base em histórico e condições ambientais.
2. O uso de sensores de corrente calibrados e registro em tempo real para obtenção de curvas de consumo experimentais e validação prática do modelo teórico.

Em síntese, o artefato proposto atingiu seus objetivos de projeto, comprovando que a adoção de estratégias de gerenciamento adaptativo de energia contribui de forma significativa para a autonomia, eficiência e sustentabilidade de sistemas de monitoramento baseados em redes de sensores.

8. Referências

AKYILDIZ, I. F.; SU, W.; SANKARASUBRAMANIAM, Y.; CAYIRCI, E. Wireless sensor networks: a survey. *Computer Networks*, v. 38, n. 4, p. 393–422, 2002. DOI: 10.1016/S1389-1286(01)00302-4. Disponível em: https://doi.org/10.1016/S1389-1286(01)00302-4.

BALBINOT, A.; BRUSAMARELLO, V. *Instrumentação e fundamentos de medidas*. Rio de Janeiro: LTC, 2011. (v. 2). ISBN 978-85-216-1879-9. Disponível em: https://www.booki.pt/loja/prod/instrumentacao-e-fundamentos-de-medidas-vol-2/9788521618799/.

RUIZ, L. B.; CORREIA, L. H.; VIEIRA, L. F. M. *et al.* Arquiteturas para Redes de Sensores Sem Fio. In: SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES (SBRC), 22., 2004, Gramado. *Minicursos*. Gramado: SBC, 2004. p. 167–218. ISBN

85-88442-81-7. Disponível em: https://www.dcc.ufmg.br/~mmvieira/publications/04mc-sbrc.pdf.

NATIONAL RESEARCH COUNCIL. Technology for the United States Navy and Marine Corps, 2000–2035: Becoming a 21st-Century Force: Volume 2: Technology. Washington, DC: National Academies Press, 1997. DOI: 10.17226/5863. Disponível em: https://doi.org/10.17226/5863. Acesso em: 8 nov. 2025.

FÖRSTER, A. Introduction to Wireless Sensor Networks. Hoboken, NJ: Wiley-IEEE Press, 2016. Print ISBN: 978-1-118-99351-4; Online ISBN: 978-1-119-34534-3. Disponível em: https://onlinelibrary.wiley.com/doi/book/10.1002/9781119345343.

GARCÍA-HERNÁNDEZ, C. F.; IBARGÜENGOYTIA-GONZÁLEZ, P. H.; GARCÍA-HERNÁNDEZ, J.; PÉREZ-DÍAZ, J. A. Wireless Sensor Networks and Applications: a Survey. IJCSNS – International Journal of Computer Science and Network Security, v. 7, n. 3, p. 264–273, 2007. Disponível em: https://www.academia.edu/8659064/Wireless_Sensor_Networks_and_Applications_a_Survey.

GOLDEMBERG, J.; LUCON, O. Energias renováveis: um futuro sustentável. *Revista USP*, n. 72, p. 6–15, 2007. DOI: 10.11606/issn.2316-9036.v0i72p6-15. Disponível em: https://revistas.usp.br/revusp/article/view/13564.

HEINZELMAN, W. B.; CHANDRAKASAN, A.; BALAKRISHNAN, H. Energy-efficient communication protocol for wireless microsensor networks. In: *HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES (HICSS)*, 33., 2000, Maui. *Proceedings…* Los Alamitos, CA: IEEE Computer Society, 2000. p. 1–10. Disponível em: https://pdos.csail.mit.edu/archive/decouto/papers/heinzelman00.pdf.

LOUREIRO, A. A. F.; NOGUEIRA, J. M. S.; RUIZ, L. B.; MINI, R. A. F.; NAKAMURA, E. F.; FIGUEIREDO, C. M. S. Redes de sensores sem fio. In: *SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES (SBRC)*, 21., 2003, Natal. *Anais…* Natal: SBC, 2003. p. 179–226. Disponível em: https://www.dcc.ufmg.br/~LOUREIRO/cm/docs/sbrc03.pdf.

MOLLA, M. M.; AHAMED, S. I. A survey of middleware for wireless sensor networks. In: **INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING WORKSHOPS (ICPPW)**, 2006, Columbus, OH. *Proceedings…* Washington, DC: IEEE Computer Society, 2006. p. 223–228. DOI: 10.1109/ICPPW.2006.18. Disponível em: https://doi.org/10.1109/ICPPW.2006.18.

POLASTRE, J.; HILL, J.; CULLER, D. Versatile low power media access for wireless sensor networks. In: **INTERNATIONAL CONFERENCE ON EMBEDDED NETWORKED SENSOR SYSTEMS (SENSYS)**, 2., 2004, Baltimore. *Proceedings…* New York: ACM, 2004. p. 95–107. DOI: 10.1145/1031495.1031508. Disponível em: https://doi.org/10.1145/1031495.1031508.

RAZZAQUE, M. A.; HARJANI, J. **Energy-efficient design for IoT-enabled wireless sensor networks**. IEEE Internet of Things Journal, v. 4, n. 5, p. 1529–1542, 2017.

RUIZ, L. B.; NOGUEIRA, J. M. S.; LOUREIRO, A. A. F. MANNA: a management architecture for wireless sensor networks. *IEEE Communications Magazine*, v. 41, n. 2, p. 116–125, 2003. DOI: 10.1109/MCOM.2003.1179560. Disponível em: https://doi.org/10.1109/MCOM.2003.1179560.

RUIZ, L. B.; CORREIA, L. H. A.; VIEIRA, L. F. M.; MACEDO, D. F.; NAKAMURA, E. F.; FIGUEIREDO, C. M. S.; VIEIRA, M. A. M.; MECHELANE, E. H.; CAMARA, D.; LOUREIRO, A. A. F.; NOGUEIRA, J. M. S.; SILVA JR., D. C. Arquiteturas para Redes de Sensores Sem Fio. In: **SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES (SBRC)**, 22., 2004, Gramado. *Minicursos*. Gramado: SBC, 2004. p. 167–218. ISBN 85-88442-81-7. Disponível em: https://www.dcc.ufmg.br/~mmvieira/publications/04mc-sbrc.pdf.

SOUSA, J. R. B.; LIMA, A. M. N.; SAUSEN, P. S.; PERKUSICH, A. Redes de Petri híbridas diferenciais: aplicação na modelagem e no gerenciamento dinâmico de energia de redes de sensores sem fio. *Controle & Automação*, v. 18, n. 3, p. 278–291, 2007. DOI: 10.1590/S0103-17592007000300002. Disponível em: https://doi.org/10.1590/S0103-17592007000300002.

TEIXEIRA, M. M. Transdutor. *Brasil Escola*, [s. l.], [s. d.]. Disponível em: https://brasilescola.uol.com.br/fisica/transdutor.htm.

TIWARI, P.; SAXENA, V. P.; MISHRA, R. G.; BHAVSAR, D. Wireless Sensor Networks: Introduction, Advantages, Applications and Research Challenges. *HCTL Open International Journal of Technology Innovations and Research*, v. 14, 2015. Disponível em: https://www.researchgate.net/publication/296802403_Wireless_Sensor_Networks_Introduction_Advantages_Applications_and_Research_Challenges.

VIDHYAPRIYA, R.; VANATHI, P. T. Energy-aware routing for wireless sensor networks. In: *INTERNATIONAL CONFERENCE ON SIGNAL PROCESSING, COMMUNICATIONS AND NETWORKING (ICSCN)*, 2007, Chennai. *Proceedings…*

IEEE, 2007. p. 545–550. Disponível em: https://www.researchgate.net/profile/R-Vidhyapriya/publication/228411173_Energy_Aware_Routing_for_Wireless_Sensor_Networks/links/5521f8b50cf29dcabb0d2fdd/Energy-Aware-Routing-for-Wireless-Sensor-Networks.pdf.

YE, W.; HEIDEMANN, J.; ESTRIN, D. An energy-efficient MAC protocol for wireless sensor networks. In: *IEEE INFOCOM*, 21., 2002, New York. *Proceedings…* IEEE, 2002. p. 1567–1576. DOI: 10.1109/INFCOM.2002.1019408. Disponível em: https://doi.org/10.1109/INFCOM.2002.1019408.

YICK, J.; MUKHERJEE, B.; GHOSAL, D. Wireless sensor network survey. *Computer Networks*, v. 52, n. 12, p. 2292–2330, 2008. DOI: 10.1016/j.comnet.2008.04.002. Disponível em: https://dl.acm.org/doi/10.1016/j.comnet.2008.04.002.

APÊNDICE A – Código-fonte do Arduino (Módulo Sensor e Controle de Energia)

```
#include <DHT.h>
#include <avr/sleep.h>
#include <avr/power.h>

#define PIN_VBAT A0
#define US_TRIG 7
#define US_ECHO 6
#define PIR_PIN 5
#define DHTPIN 8
#define DHTTYPE DHT11
#define LDR_PIN A1
#define MQ2_PIN A2

#define PWR_DHT 22
#define PWR_LDR 23
#define PWR_PIR 24
#define PWR_US 25
#define PWR_MQ2 26

const bool HAS_PWR_CUT_DHT = true;
const bool HAS_PWR_CUT_LDR = true;
```

```
const bool HAS_PWR_CUT_PIR = true;
const bool HAS_PWR_CUT_US = true;
const bool HAS_PWR_CUT_MQ2 = true;

DHT dht(DHTPIN, DHTTYPE);

float VREF = 5.00;
float FACTOR = 5.00;
const float VMAX = 12.0;
const float VMIN = 7.0;
const float PMIN = 5.0;
const int HYST = 2;

enum Mode { MODE_RESERVA, MODE_CRITICO, MODE_ECONOMIA, MODE_NORMAL };
struct Policy { bool dht_on, ldr_on, pir_on, us_on, mq2_on; unsigned long
read_interval_ms; };

const Policy POL_RESERVA = { true, true, false, false, false, 30000 };
const Policy POL_CRITICO = { true, true, true, false, false, 5000 };
const Policy POL_ECONOMIA = { true, true, true, false, false, 3500 };
const Policy POL_NORMAL = { true, true, true, true, true, 2000 };

const unsigned long MQ2_WARMUP_MS = 60000;
const unsigned long MQ2_BASE_MS = 5000;
const float MQ2_EMA_ALPHA = 0.2;
bool mq2_ready = false, mq2_warming = true, mq2_basing = false;
unsigned long mq2_t0 = 0;
int mq2_base = 0;
float mq2_ema = 0;

volatile bool MODE_OVERRIDE = false;
volatile Mode forcedMode = MODE_NORMAL;
Mode curMode = MODE_NORMAL;
unsigned long lastRead = 0;

float readVbat() {
  long s = 0;
  for (int i = 0; i < 40; i++) { s += analogRead(PIN_VBAT); delay(2); }
  float adc = s / 40.0;
  float vadc = (adc / 1023.0) * VREF;
  return vadc * FACTOR;
```

```
}

int pctFromV(float v) {
  if (v <= VMIN) return (int)PMIN;
  if (v >= VMAX) return 100;
  float p = PMIN + (v - VMIN) * (100.0 - PMIN) / (VMAX - VMIN);
  if (p < PMIN) p = PMIN; if (p > 100) p = 100;
  return (int)(p + 0.5f);
}

const char* faixa(Mode m) {
  switch (m) {
    case MODE_NORMAL: return "normal";
    case MODE_ECONOMIA: return "economia";
    case MODE_CRITICO: return "critico";
    default: return "reserva";
  }
}

float readUS() {
  digitalWrite(US_TRIG, LOW); delayMicroseconds(2);
  digitalWrite(US_TRIG, HIGH); delayMicroseconds(10);
  digitalWrite(US_TRIG, LOW);
  long d = pulseIn(US_ECHO, HIGH, 30000);
  if (d <= 0) return -1;
  return d * 0.0343f / 2.0f;
}

int readPIR() { return digitalRead(PIR_PIN); }
int readLDR() { long s = 0; for (int i = 0; i < 10; i++) { s +=
analogRead(LDR_PIN); delay(2);} return (int)(s / 10); }
int readMQ2Raw() { long s = 0; for (int i = 0; i < 20; i++) { s +=
analogRead(MQ2_PIN); delay(2);} return (int)(s / 20); }

int mq2Percent(int raw) {
  if (!mq2_ready) return -1;
  long num = (long)raw - (long)mq2_base; if (num < 0) num = 0;
  long den = 1023 - mq2_base; if (den <= 0) den = 1;
  long pct = (num * 100L) / den; if (pct > 100) pct = 100; return (int)pct;
}
```

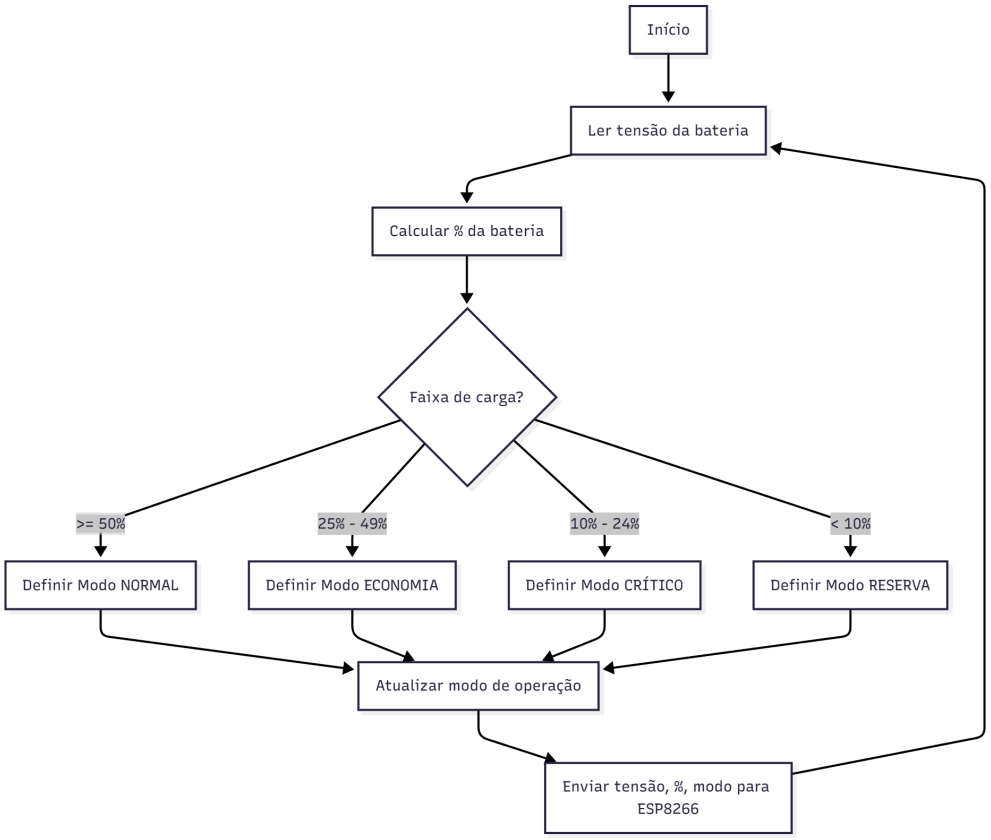
```
void mq2Recalibrate() { mq2_warming = false; mq2_basing = true; mq2_ready = false; mq2_t0 = millis(); mq2_base = readMQ2Raw(); mq2_ema = mq2_base; }
void mq2Begin() { mq2_t0 = millis(); mq2_warming = true; mq2_basing = false; mq2_ready = false; }

void mq2Task() {
  if (mq2_warming) { if (millis() - mq2_t0 >= MQ2_WARMUP_MS) { mq2_basing = true; mq2_warming = false; mq2_t0 = millis(); mq2_base = readMQ2Raw(); mq2_ema = mq2_base; } return; }
  if (mq2_basing) { if (millis() - mq2_t0 <= MQ2_BASE_MS) { static long acc = 0; static int n = 0; acc += readMQ2Raw(); n++; mq2_base = (int)(acc / (long)n); }
    else { mq2_basing = false; mq2_ready = true; } return; }
    int raw = readMQ2Raw(); mq2_ema = MQ2_EMA_ALPHA * raw + (1.0 - MQ2_EMA_ALPHA) * mq2_ema;
}

void setPower(uint8_t pin, bool on, bool wired) { if (!wired) return; digitalWrite(pin, on ? HIGH : LOW); }

void applyPolicy(const Policy& pol) {
  setPower(PWR_DHT, pol.dht_on, HAS_PWR_CUT_DHT);
  setPower(PWR_LDR, pol.ldr_on, HAS_PWR_CUT_LDR);
  setPower(PWR_PIR, pol.pir_on, HAS_PWR_CUT_PIR);
  setPower(PWR_US, pol.us_on, HAS_PWR_CUT_US);
  setPower(PWR_MQ2, pol.mq2_on, HAS_PWR_CUT_MQ2);
}

Mode decideMode(int pct, Mode prev) {
  switch (prev) {
    case MODE_NORMAL:   if (pct <= 50 - HYST) return MODE_ECONOMIA; return MODE_NORMAL;
    case MODE_ECONOMIA: if (pct >= 50 + HYST) return MODE_NORMAL; if (pct <= 25 - HYST) return MODE_CRITICO; return MODE_ECONOMIA;
    case MODE_CRITICO:  if (pct >= 25 + HYST) return MODE_ECONOMIA; if (pct <= 10 - HYST) return MODE_RESERVA; return MODE_CRITICO;
    case MODE_RESERVA:  if (pct >= 10 + HYST) return MODE_CRITICO; return MODE_RESERVA;
  } return prev;
}
```

```
void enterDeepSleep() {
  set_sleep_mode(SLEEP_MODE_PWR_DOWN);
  sleep_enable();
  sleep_cpu();
}

void notifyESP_Mode(Mode m) {
  if (m == MODE_ECONOMIA || m == MODE_CRITICO) {
    Serial1.println("#SLEEP_MODE");
  } else {
    Serial1.println("#WAKE_MODE");
  }
}

void setup() {
  Serial.begin(9600);
  Serial1.begin(9600);
  pinMode(US_TRIG, OUTPUT); pinMode(US_ECHO, INPUT);
  pinMode(PIR_PIN, INPUT);
  dht.begin();
  pinMode(PWR_DHT, OUTPUT); pinMode(PWR_LDR, OUTPUT);
     pinMode(PWR_PIR,  OUTPUT);  pinMode(PWR_US,  OUTPUT);  pinMode(PWR_MQ2,
OUTPUT);
  applyPolicy(POL_NORMAL);
  mq2Begin();
}

void loop() {
  if (Serial.available()) {
    char c = Serial.read();
    if (c == 'O' || c == 'o') { MODE_OVERRIDE = !MODE_OVERRIDE; }
    if (c == 'N' || c == 'n') { forcedMode = MODE_NORMAL; }
    if (c == 'E' || c == 'e') { forcedMode = MODE_ECONOMIA; }
    if (c == 'C' || c == 'c') { forcedMode = MODE_CRITICO; }
    if (c == 'R' || c == 'r') { forcedMode = MODE_RESERVA; }
  }

  if (Serial1.available()) {
    char c = Serial1.read();
    if (c == 'N') forcedMode = MODE_NORMAL;
    if (c == 'E') forcedMode = MODE_ECONOMIA;
```

```
    if (c == 'C') forcedMode = MODE_CRITICO;
    if (c == 'R') forcedMode = MODE_RESERVA;
    MODE_OVERRIDE = true;
  }

  mq2Task();
  float vbat = readVbat();
  int pct = pctFromV(vbat);
  Mode next = MODE_OVERRIDE ? forcedMode : decideMode(pct, curMode);

  if (next != curMode) {
    curMode = next;
    switch (curMode) {
      case MODE_NORMAL:   applyPolicy(POL_NORMAL); break;
      case MODE_ECONOMIA: applyPolicy(POL_ECONOMIA); break;
      case MODE_CRITICO:  applyPolicy(POL_CRITICO); break;
      case MODE_RESERVA:  applyPolicy(POL_RESERVA); break;
    }
    notifyESP_Mode(curMode);
    if (curMode == MODE_RESERVA) {
      Serial.println("Entrando em deep sleep...");
      delay(100);
      enterDeepSleep();
    }
  }

  const Policy* P = &POL_NORMAL;
  switch (curMode) {
    case MODE_NORMAL:   P = &POL_NORMAL; break;
    case MODE_ECONOMIA: P = &POL_ECONOMIA; break;
    case MODE_CRITICO:  P = &POL_CRITICO; break;
    default:            P = &POL_RESERVA; break;
  }

  if (millis() - lastRead >= P->read_interval_ms) {
    lastRead = millis();
    float cm = -1;  if (P->us_on)  cm = readUS();
    int pir = -1;   if (P->pir_on) pir = readPIR();
    int ldr = -1;   if (P->ldr_on) ldr = readLDR();
     float t = NAN, h = NAN; if (P->dht_on) { t = dht.readTemperature(); h =
dht.readHumidity(); }
```

```
    int mq2_raw = -1, mq2_pct = -1; if (P->mq2_on) { mq2_raw = (int)mq2_ema;
mq2_pct = mq2Percent(mq2_raw); }

    String data = String("V=") + String(vbat, 2) +
                  " P=" + String(pct) + "%" +
                  " " + faixa(curMode) +
                  " | U=" + (cm < 0 ? String("--") : String(cm, 1)) + "cm" +
                  " | PIR=" + (pir < 0 ? String("--") : String(pir)) +
                  " | L=" + (ldr < 0 ? String("--") : String(ldr)) +
                  " | T=" + (isnan(t) ? String("--") : String(t, 1)) +
                  " H=" + (isnan(h) ? String("--") : String(h, 1)) +
                  " | MQ2=" + (mq2_raw < 0 ? String("--") : String(mq2_raw));

    Serial.println(data);
            Serial1.write('^');   Serial1.print(data);   Serial1.write('$');
Serial1.write('\n');
  }
```

Fluxograma simplificado do código



APÊNDICE B – Código-fonte do ESP8266 (Comunicação e Interface Web)

```
#include <ESP8266WiFi.h>
```

```
#include <ESP8266WebServer.h>
#include <WiFiClientSecure.h>
#include <ESP8266HTTPClient.h>

const char* ssid = "vitor";
const char* password = "vitor123";
const char* serverUrl =
"https://silver-fox-585643.hostingersite.com/receive.php";

ESP8266WebServer server(80);

String urlencode(const String& s) {
  String o; const char* h = "0123456789ABCDEF";
  for (size_t i = 0; i < s.length(); i++) {
    unsigned char c = s[i];
    if (isalnum(c) || c == '-' || c == '_' || c == '.' || c == '~') o +=
char(c);
    else if (c == ' ') o += '+';
    else { o += '%'; o += h[(c >> 4) & 0xF]; o += h[c & 0xF]; }
  }
  return o;
}

bool readFrame(String& out) {
  while (Serial.available()) {
    if (Serial.read() == '^') break;
  }
  unsigned long t0 = millis();
  String s; s.reserve(128);
  while (millis() - t0 < 400) {
    while (Serial.available()) {
      char c = Serial.read();
      if (c == '$') { out = s; out.trim(); return out.length() > 0; }
      if (c != '\r' && c != '\n') s += c;
    }
    yield();
  }
  return false;
}

void handleRoot() {
```

```
  server.send(200, "text/plain", "ESP8266 Online");
}

void handleSetMode() {
  if (!server.hasArg("mode")) {
    server.send(400, "application/json", "{\"error\":\"missing mode\"}");
    return;
  }

  String mode = server.arg("mode");
  mode.toLowerCase();
  char cmd = 0;

  if (mode == "normal") cmd = 'N';
  else if (mode == "economia") cmd = 'E';
  else if (mode == "critico") cmd = 'C';
  else if (mode == "reserva") cmd = 'R';

  if (cmd) {
    Serial.write(cmd);
    server.send(200, "application/json", "{\"status\":\"ok\",\"mode\":\"" +
mode + "\"}");
  } else {
    server.send(400, "application/json", "{\"error\":\"invalid mode\"}");
  }
}

void setup() {
  Serial.begin(9600);
  Serial.setTimeout(50);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(200);
  }

  server.on("/", handleRoot);
  server.on("/setmode", handleSetMode);
  server.begin();
```

```
  Serial.println();
  Serial.print("ESP8266 IP: ");
  Serial.println(WiFi.localIP());
}

void loop() {
  server.handleClient();

  if (WiFi.status() != WL_CONNECTED) {
    WiFi.begin(ssid, password);
    delay(600);
    return;
  }

  String data;
  if (readFrame(data)) {
    WiFiClientSecure client; client.setInsecure();
    HTTPClient http; http.setTimeout(8000);
    if (http.begin(client, serverUrl)) {
      http.addHeader("Content-Type", "application/x-www-form-urlencoded");
      http.addHeader("Connection", "close");
      int code = http.POST(String("dados=") + urlencode(data));
      http.end();
    }
  }
}
```

APÊNDICE C – Detalhamento do Hardware e Mapeamento de Pinos

Este apêndice detalha os componentes de hardware utilizados no protótipo do nó sensor e especifica o mapeamento de pinos do microcontrolador Arduino Mega 2560. A configuração de hardware é a base para a implementação do software de gerenciamento de energia descrito no Capítulo 5.

1. Componentes Principais

- Microcontrolador: Arduino Mega 2560;
- Módulo de Comunicação: ESP8266;
- Sensor de tensão (para leitura da Vbat);
- Sensores:

- DHT11;
- LDR;
- PIR HC-SR501;
- HC-SR04;
- MQ-2.

2. Mapeamento de Pinos

Componente	Função	Pino no Arduino	Tipo de Pino
Sensor de Tensão	Leitura Vbat	A0	Analógico
DHT11	Sinal (Dados)	8	Digital
	Controle de Energia	22	Digital (Saída)
LDR	Sinal (Dados)	A1	Analógico
	Controle de Energia	23	Digital (Saída)
PIR HC-SR501	Sinal (Dados)	5	Digital
	Controle de Energia	24	Digital (Saída)
HC-SR04	Sinal (Trigger)	7	Digital (Saída)
	Sinal (Echo)	6	Digital (Entrada)
	Controle de Energia	25	Digital (Saída)
MQ-2	Sinal (Dados)	A2	Analógico
	Controle de	26	Digital (Saída)

	Energia		
ESP8266	Comunicação (TX)	TX1 (Pino 18)	Serial Hardware
	Comunicação (RX)	RX1 (Pino 19)	Serial Hardware