Proposta de módulo de comunicação para as estações meteorológicas do sistema EPAGRI/CIRAM

João Pedro Tietbohl[1], **Tales Reig Cordova**[1], **Robson Costa**[1]

[1]Instituto Federal de Santa Catarina (IFSC)
Rua Heitor Vila Lobos, 225 – Bairro São Francisco – 88506-400 – Lages – SC – Brasil

joao.tc12@aluno.ifsc.edu.br, tales.rc13@aluno.ifsc.edu.br

robson.costa@ifsc.edu.br

Abstract. *In the face of increasing climate change, meteorological stations in the EPAGRI/CIRAM system play a fundamental role in environmental monitoring, preventing natural disasters and supporting sustainable agricultural practices, making it essential to ensure their up-to-date operation to obtain accurate data and protect communities. This work presents the development of a communication module that modernizes these stations, replacing obsolete technologies such as 2G and 3G with modern protocols such as MQTT and 4G to collect data from sensors in order to guarantee efficient and reliable transmission.*

Resumo. *Diante das crescentes mudanças climáticas, as estações meteorológicas do sistema EPAGRI/CIRAM desempenham um papel fundamental no monitoramento ambiental, na prevenção de desastres naturais e no apoio a práticas agrícolas sustentáveis, tornando essencial garantir seu funcionamento atualizado para obter dados precisos e proteger comunidades. Este trabalho apresenta o desenvolvimento de um módulo de comunicação que moderniza essas estações, substituindo tecnologias obsoletas como 2G e 3G por protocolos modernos como MQTT e 4G para a coleta de dados dos sensores de forma a garantir, assim, uma transmissão eficiente e confiável.*

1. Introdução

A Internet das Coisas (do inglês – *Internet of Things* - IoT) é considerado o paradigma computacional que mais cresceu desde a sua criação devido ao impacto que tem no campo da tecnologia, permitindo a comunicação inteligente entre dispositivos e sistemas em escala global (Abdul Hafeez et al., 2022). A coleta e o compartilhamento de dados em tempo real têm possibilitado sua aplicação em diversos setores, desde a indústria até a saúde, alterando significativamente a interação com o ambiente e otimizando processos.

A aplicação da IoT em ambientes agrícolas e de monitoramento ambiental tem se destacado pela sua capacidade de transformar a gestão dos recursos naturais e a produção agrícola. Através da utilização de sensores inteligentes e dispositivos conectados, a IoT permite a coleta detalhada de dados sobre condições climáticas, qualidade do solo, níveis de água, entre outros parâmetros ambientais. Esses dados são analisados em tempo real, proporcionando aos produtores agrícolas e gestores ambientais uma tomada de decisão mais precisa e fundamentada. Além disso, a IoT facilita a implementação de práticas de agricultura de precisão, como o uso eficiente de recursos hídricos e insumos agrícolas,

monitoramento do crescimento das culturas e a detecção precoce de pragas e doenças. A agricultura de precisão, por meio da IoT, tem promovido maior eficiência na utilização dos recursos e melhorado a sustentabilidade das práticas agrícolas (Ayaz et al., 2019). A IoT também tem desempenhado um papel crucial no desenvolvimento de tecnologias emergentes para a agricultura inteligente, como sensores e drones para monitoramento ambiental, possibilitando a automação das práticas agrícolas e a otimização no uso de água e fertilizantes (Pyingkodi et al., 2022).

Essas mesmas tecnologias também têm sido empregadas na prevenção de desastres naturais, como enchentes, deslizamentos de terra e terremotos. Sensores inteligentes conectados à Internet possibilitam a coleta e análise em tempo real de dados ambientais essenciais, capacitando as autoridades a emitirem alertas precoces e tomarem medidas proativas para proteger comunidades vulneráveis. As tecnologias baseadas em IoT têm-se mostrado cruciais na gestão de desastres, permitindo sistemas de alerta precoce e melhorando a resiliência das comunidades afetadas (Rajendran et al., 2023). Essa integração em sistemas de monitoramento ambiental demonstra seu potencial em mitigar riscos e promover a resiliência em face de eventos climáticos extremos.

No Brasil, instituições como o Centro Nacional de Monitoramento e Alertas de Desastres Naturais (CEMADEN) e a Empresa de Pesquisa Agropecuária e Extensão Rural de Santa Catarina (EPAGRI) através do Centro de Informações de Recursos Ambientais e de Hidrometeorologia de Santa Catarina (CIRAM) têm liderado esforços inovadores na agricultura de precisão e na prevenção de desastres ambientais. Essas entidades visam maximizar a eficiência agrícola e enviar alertas para as comunidades sobre os riscos de possíveis desastres ambientais. Neste contexto, a EPAGRI/CIRAM, através do projeto *Agroconnect*[1] realiza o monitoramento climático em tempo real e gera alertas precoces de eventos climáticos extremos.

Em um ambiente de avanço tecnológico e ênfase na utilização da IoT, torna-se imprescindível melhorar a infraestrutura de comunicação, a fim de acompanhar o ritmo da evolução tecnológica e viabilizar a sua utilização. No caso específico das estações meteorológicas da EPAGRI/CIRAM, essa necessidade de atualização é ainda mais urgente. Os equipamentos em uso nessas estações empregam tecnologias antiquadas, como 2G e 3G, as quais se encontram em processo de descontinuação (ANATEL, 2024).

Diante desse cenário, o objetivo geral deste trabalho é desenvolver um módulo de comunicação que atenda às necessidades específicas das estações meteorológicas da EPAGRI/CIRAM. Esse módulo deverá ser capaz de extrair os dados armazenados nos *dataloggers* presentes nas estações e transmiti-los de forma eficiente e confiável para o servidor da instituição, além de também permitir a parametrização e atualização do *firmware* remotamente.

Para atingir o objetivo geral, os seguintes objetivos específicos foram definidos:

- Realizar o levantamento de requisitos;
- Projetar uma *Printed Circuit Board* (PCB) utilizando elementos de *hardware* que melhor se adéquem aos requisitos do projeto;
- Desenvolver o *firmware* a ser utilizado no dispositivo;
- Executar testes operacionais de uma unidade piloto em um ambiente real;

[1] `https://ciram.epagri.sc.gov.br/agroconnect/`

A metodologia adotada neste trabalho foi de natureza aplicada, abordando o problema de forma mista combinando elementos quantitativos e qualitativos. O estudo foi conduzido com uma abordagem exploratória-descritiva, visando entender e descrever fenômenos específicos, e empregando procedimentos técnicos de pesquisa-ação, sendo possível dividir o estudo em três etapas (Silva e Menezes, 2005). Na primeira etapa, foi realizado um estudo dos requisitos técnicos e tecnologias envolvidas no desenvolvimento da solução proposta. A segunda etapa consistiu no desenvolvimento de *hardware* e *firmware* do dispositivo em questão. Por fim, na terceira e última etapa, foi utilizado um *middleware* para a definição e realização de um conjunto de testes em um ambiente de teste real, permitindo a análise, comparação e compreensão dos resultados obtidos.

Este documento está organizado da seguinte forma: na seção 2 é apresentado o referencial teórico utilizado como base para o desenvolvimento deste trabalho; na seção 3 é descrita e detalhada a solução proposta; na seção 4 é detalhado como foi realizado o desenvolvimento da solução com base no referencial teórico e na modelagem apresentadas anteriormente; na seção 5 são apresentados os testes realizados, bem como são discutidos os seus respectivos resultados; por fim, na seção 6, é retratada a conclusão obtida, assim como direcionamentos para trabalhos futuros.

2. Referencial Teórico

Nesta seção, foram estabelecidos os fundamentos essenciais necessários para compreender e implementar soluções que se utilizam da IoT. Os conceitos-chave que sustentam essa tecnologia revolucionária foram explorados, fornecendo uma base sólida para o desenvolvimento e compreensão de suas aplicações. Além disso, foram detalhados os principais protocolos de comunicação, com especial ênfase nas soluções de *Low Power Wide Area Network* (LPWAN), abordando suas características, benefícios e potenciais aplicações.

2.1. Internet das Coisas

A IoT representa uma revolução tecnológica que vai além do simples controle de dispositivos, como acender e apagar lâmpadas, por meio de *smartphones*. Ela envolve a conexão e a inteligência desses dispositivos, permitindo que coletem e processem informações do ambiente ou das redes às quais estão conectados (Oliveira, 2017). Esta transformação está redefinindo a maneira como se interage com o ambiente, possibilitando a coleta e o compartilhamento de dados em tempo real sem a necessidade de intervenção humana direta, o que tem implicações significativas em diversos setores.

Após compreender esse conceito, é essencial explorar as principais vantagens que têm impulsionado a ampla adoção da IoT em diversos setores. De acordo com Fetahu et al. (2022), as principais vantagens do uso da IoT incluem:

1. **Coleta de Dados em Tempo Real**: Sensores e dispositivos conectados fornecem dados em tempo real, permitindo tomadas de decisão mais informadas e precisas.
2. **Automatização e Eficiência Operacional**: A IoT permite a automação de processos e operações, aumentando a eficiência e reduzindo custos.
3. **Otimização de Recursos**: A IoT ajuda na gestão inteligente de recursos, como energia, água e transporte, contribuindo para a sustentabilidade ambiental.
4. **Integração e Conectividade**: A capacidade de conectar diversos dispositivos e sistemas promove uma maior integração e colaboração entre diferentes plataformas e serviços.

No entanto, como também destacado por Fetahu et al. (2022), é igualmente importante considerar os desafios que acompanham a implementação da IoT:

1. **Segurança e Privacidade**: A expansão da IoT aumenta os riscos de vulnerabilidades de segurança e violações de privacidade, exigindo medidas robustas de proteção.
2. **Sobrecarga de Dados**: A enorme quantidade de dados gerados pela IoT pode sobrecarregar os sistemas de armazenamento e análise, exigindo soluções escaláveis e eficientes.
3. **Dependência de Conectividade**: A operação eficaz da IoT depende de uma conexão de rede estável e confiável, o que pode ser um desafio em áreas remotas ou com infraestrutura de rede inadequada.

Em resumo, a IoT oferece benefícios significativos em termos de automação, eficiência e melhorias na qualidade de vida, mas também apresenta desafios que precisam ser abordados para garantir sua adoção e implementação bem-sucedidas.

Por conta disso, o uso da IoT está crescendo em diversos setores. Entre eles, o setor agrícola e ambiental vem se destacando, utilizando a tecnologia para melhorar a eficiência de suas operações, monitorar condições de campo e otimizar o uso de recursos naturais. Com a constante evolução da tecnologia e a crescente demanda por soluções sustentáveis, espera-se que a IoT continue desempenhando um papel fundamental na transformação digital desses setores.

2.1.1. Aplicações da IoT na Agricultura

A agricultura passou por diversas transformações ao longo dos anos (Figura 1), desde os métodos tradicionais até as práticas mais modernas e tecnológicas. No entanto, por volta dos anos 2000, surgiu uma nova era na agricultura: as *SmartFarms* ou fazendas inteligentes.

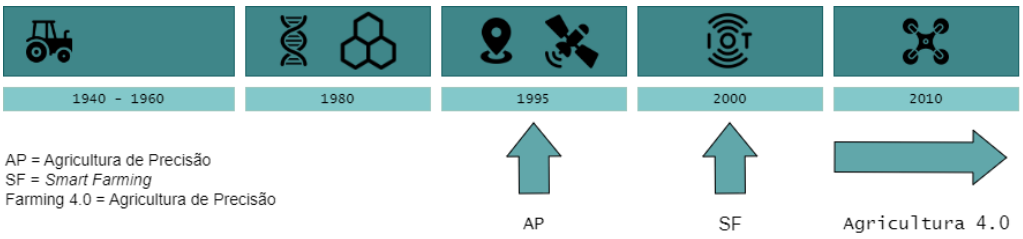

Figura 1. Linha do tempo das tecnologias na agricultura, adaptada de Junior e Luiz (2018).

É importante destacar a diferença entre agricultura de precisão, *SmartFarms* e agricultura 4.0. Enquanto a agricultura de precisão se concentra na aplicação precisa de insumos agrícolas possibilitadas pelo desenvolvimento do GPS, o *SmartFarms* busca otimizar as operações agrícolas com a ajuda de tecnologias avançadas, como a IoT. Por sua vez, a agricultura 4.0 representa uma visão mais ampla e integrada, transformando toda a cadeia de produção agrícola através da digitalização e automação.

As *SmartFarms* têm transformado profundamente o cenário agrícola, incorporando tecnologias de ponta como IoT, Inteligência Artificial (IA) e *drones*. Estas inovações possibilitam uma gestão agrícola mais eficiente, oferecendo monitoramento contínuo das condições das lavouras e rebanhos, além de facilitar a tomada de decisões embasadas em dados precisos e em tempo real. O foco dos estudos em agricultura inteligente é estabelecer uma base para um sistema de suporte à tomada de decisão de gestão agrícola (Karthiga et al., 2023).

Atualmente, em alguns ambientes de produção agrícola, já são empregadas técnicas de alta precisão, a fim de melhorar a eficiência da produção e das operações diárias. Por exemplo, os sensores colocados nos campos permitem que os agricultores obtenham mapas detalhados da topografia e dos recursos da área, bem como variáveis como a acidez e a temperatura do solo. Eles também podem acessar previsões meteorológicas para prevenir-se de padrões climáticos nos próximos dias e semanas. Há uma grande quantidade de dados provenientes de diferentes fontes, como equipamentos de irrigação, fertilizantes, semeadores e sensores de umidade no solo, e esses dados são integrados e analisados de forma conjunta, permitindo aos agricultores tomar decisões mais informadas e precisas (Junior e Luiz, 2018).

Em resumo, a integração da IoT na agricultura está revolucionando a maneira como os agricultores operam, permitindo uma produção agrícola mais inteligente, otimizada e resiliente às mudanças climáticas e desafios ambientais.

2.1.2. Aplicações da IoT no Monitoramento Ambiental

A IoT também pode ser utilizada no monitoramento ambiental e na prevenção de desastres naturais, oferecendo soluções inovadoras para enfrentar os desafios das mudanças climáticas e eventos extremos. Sensores conectados possibilitam a coleta de dados em tempo real sobre condições climáticas, qualidade do ar e níveis de água, possibilitando respostas rápidas a emergências. Esses sistemas auxiliam na detecção precoce de mudanças ambientais, alertando sobre eventos climáticos extremos e contribuindo para a preservação de recursos hídricos e a segurança das comunidades.

Segundo Kuo et al. (2022), a integração da IoT ao monitoramento ambiental possibilita o desenvolvimento de sistemas avançados de prevenção de desastres, permitindo a coleta em tempo real de dados ambientais cruciais. Essa abordagem não apenas proporciona alertas antecipados e precisos sobre condições climáticas extremas, mas também facilita o processo de tomada de decisão para proteger o meio ambiente e garantir a segurança das comunidades.

Outro exemplo citado por Kadir et al. (2018) é que a IoT pode ser utilizada para prevenir incêndios em áreas florestais, oferecendo uma solução eficiente e inteligente por meio de sistemas avançados e conectados de monitoramento. Esses sistemas permitem a detecção precoce dos incêndios, contribuindo para a preservação ambiental e a segurança das comunidades afetadas.

Em resumo, a IoT tem facilitado o monitoramento ambiental e a prevenção de desastres naturais, oferecendo soluções inteligentes e eficientes para proteger o meio ambiente e garantir a segurança das comunidades em todo o mundo.

2.2. Arquitetura da Internet das Coisas

Conforme destacado pelo artigo *The key layers of IoT architecture* (Bouaouad et al., 2020) quando se trata da arquitetura da IoT, ainda não existe um consenso geral sobre uma arquitetura padrão. A análise das diferentes propostas revela uma variedade significativa de números de camadas, sendo que os números mais comuns de configurações de camadas variam entre três e seis. A falta de uma arquitetura universalmente aceita reflete a natureza emergente e em constante evolução da IoT. Neste trabalho, optou-se por utilizar a arquitetura de cinco camadas proposta por Khan et al. (2012) (Figura 2) como base para análise e desenvolvimento dos aspectos arquiteturais da IoT.

Figura 2. Arquitetura da IoT, adaptada de Khan et al. (2012).

2.2.1. Camada de Percepção

A Camada de Percepção representa a interface direta e física entre o ambiente real e o ecossistema da IoT. Nesta camada, os dispositivos sensores e atuadores desempenham um papel fundamental na coleta e interação com dados do ambiente. Sensores de diferentes tipos, como temperatura, umidade, localização geográfica, movimento e outros, são utilizados para capturar informações específicas do ambiente.

Além disso, é nesta camada que se encontra o *firmware*. O *firmware* é o software ou código integrado diretamente ao *hardware* dos dispositivos. Ele é responsável por controlar as operações básicas dos dispositivos e gerenciar as interações com os sensores. O desenvolvimento, atualização e manutenção adequados do *firmware* são aspectos críticos que devem ser gerenciados e monitorados ao longo do ciclo de vida dos dispositivos IoT.

2.2.2. Camada de Rede

A Camada de Rede é responsável por conectar os dispositivos IoT à infraestrutura de rede e sistemas de *backend*. Utilizando tecnologias como Wi-Fi, *Bluetooth*, *Zigbee*, 4G, LoRa e protocolos como o MQTT, esta camada estabelece e gerencia as conexões de comunicação entre os dispositivos. Adicionalmente, mecanismos de segurança, como criptografia e autenticação, são implementados para proteger os dados durante a transmissão das informações coletadas pelos dispositivos IoT.

2.2.3. Camada de *Middleware*

A Camada de *middleware* desempenha um papel crucial na gestão dos serviços e na comunicação entre os dispositivos IoT. Cada dispositivo é projetado para interagir com outros que compartilham o mesmo tipo de serviço. Esta camada é responsável por gerenciar esses serviços e tem uma conexão direta com o banco de dados. Ela recebe dados da Camada de Rede, armazena-os no banco de dados e realiza o processamento de informações e a tomada de decisões automáticas com base nos resultados obtidos.

2.2.4. Camada de Aplicação

A Camada de Aplicação é responsável por fornecer interfaces de usuário, aplicativos e serviços que permitem aos usuários interagir e controlar os dispositivos IoT. Esta camada transforma os dados coletados e processados em informações úteis e aplicáveis para os usuários finais, permitindo visualizar, analisar e tomar decisões com base nos dados. Os aplicativos desenvolvidos nesta camada podem abranger diversas áreas e setores, como saúde, agricultura, automação residencial, cidades inteligentes, transporte, entre outros.

2.2.5. Camada de Negócios

A Camada de Negócios representa o nível estratégico da arquitetura da IoT, encarregada da gestão abrangente do sistema IoT, incluindo aplicações e serviços. Esta camada é responsável por desenvolver modelos de negócios, gráficos, fluxogramas e outras ferramentas de análise com base nos dados fornecidos pela Camada de Aplicação. Através da análise dos resultados e informações obtidas dos dados IoT, esta camada desempenha um papel crucial na definição de estratégias futuras, orientando as ações e decisões de negócio para maximizar o valor e o impacto da IoT no mercado e nas operações empresariais.

2.3. Infraestrutura de Comunicação Móvel

A evolução das tecnologias de telefonia móvel tem sido marcada por avanços significativos, cada um impulsionando a conectividade e a capacidade de comunicação. Desde o surgimento do 1G até o advento do 5G (Figura 3), testemunhamos uma transformação notável na maneira como nos comunicamos e interagimos com o mundo digital.

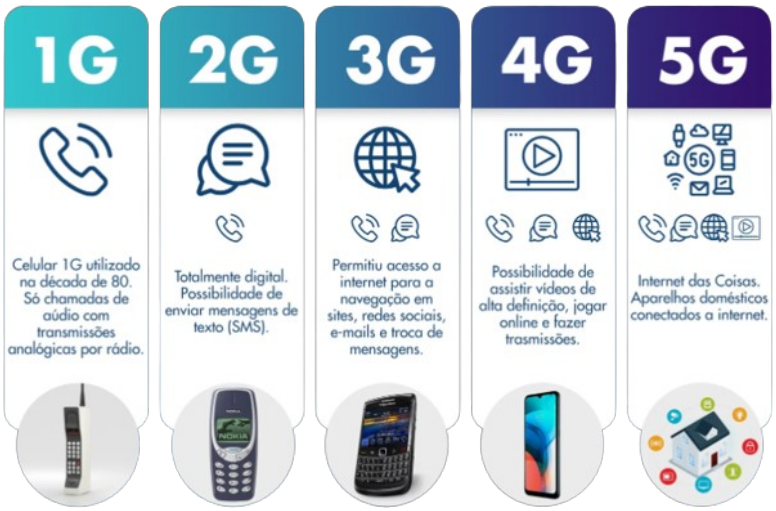

Figura 3. Evolução da comunicação, adaptada de Pinaculo (2019).

Inicialmente, com o surgimento do 1G, as comunicações foram revolucionadas, permitindo pela primeira vez chamadas de voz móveis. No entanto, a velocidade e a capacidade de transmissão de dados eram limitadas, representando apenas o primeiro passo na jornada rumo à conectividade global.

Com o 2G, a digitalização das redes proporcionou uma qualidade de chamada significativamente melhorada e a introdução de mensagens de texto. Este foi um marco importante, preparando o terreno para futuros desenvolvimentos na transmissão de dados.

O 3G marcou uma mudança fundamental, introduzindo a capacidade de transmissão de dados em velocidades mais altas. Permitiu que os *smartphones* fornecessem comunicação mais rápida, enviassem e recebessem grandes *e-mails* e textos, fornecessem navegação rápida na *web*, *streaming* de vídeo e mais segurança (Singh, 2017). Isso abriu caminho para uma variedade de serviços baseados na Internet em dispositivos móveis, permitindo o surgimento de aplicativos de mensagens. O 3G foi desenvolvido pelo *3rd Generation Partnership Project* (3GPP), uma colaboração entre várias organizações de padrões de telecomunicações, que define e supervisiona a implementação de tecnologias de telefonia móvel.

O 4G, também desenvolvido pela 3GPP, é baseado no padrão *Long Term Evolution* (LTE) e ofereceu maior largura de banda, alta segurança e acesso rápido à Internet, trazendo avanços como *streaming* de vídeo HD e Internet móvel em velocidades de até 1 *Gbps* (Odida, 2024). Sua infraestrutura robusta e madura, juntamente com sua capacidade de lidar com uma grande quantidade de conexões simultâneas, o torna ideal para aplicações IoT em diversos setores, desde saúde até agricultura e cidades inteligentes, impulsionando a inovação e a eficiência em escala global. Dentro do contexto do 4G LTE, diferentes variantes foram desenvolvidas para atender às necessidades específicas de diversos tipos de dispositivos e aplicações. Dentre elas:

- **LTE-Cat1:** Esta variante oferece conectividade de baixa potência e baixa velocidade, adequada para dispositivos que enviam pequenas quantidades de dados de forma periódica, como sensores de temperatura ou umidade em ambientes industriais ou residenciais.

- **LTE-M1:** O LTE-M1 oferece uma conectividade mais robusta e eficiente, com suporte a taxas de transferência de dados mais altas e latência reduzida. É ideal para aplicações que exigem comunicação frequente e variedade de dados, como rastreamento de ativos ou gerenciamento de frotas.
- **LTE-NB1:** Conhecido como NB-IoT, o LTE-NB1 é projetado para aplicações de de baixa potência e baixa velocidade, oferecendo uma conectividade altamente eficiente e econômica. É adequado para dispositivos que enviam pequenas quantidades de dados intermitentemente, como medidores inteligentes de energia ou sistemas de irrigação inteligente.

Todas essas variantes do 4G LTE oferecem soluções viáveis para os desafios de conectividade enfrentados pela IoT, tornando essa tecnologia ideal para uma ampla gama de aplicações.

Já o 5G é a próxima geração das comunicações móveis, oferecendo velocidades superiores a 10 Gbps, menor latência e uma capacidade de rede altamente escalável. Ele suporta uma variedade de aplicativos, desde *streaming* de vídeo 8K até realidade aumentada e virtual, além de comunicações ultra confiáveis de baixa latência e IoT em larga escala. O 5G também introduz o conceito de *network slicing*, que permite a segmentação de redes para atender a necessidades específicas de diferentes aplicativos e setores, prometendo transformar setores como saúde, manufatura e transporte.

2.4. *Modbus*

O *Modbus* é um protocolo de comunicação criado pela *Modicon* nos anos 70, amplamente utilizado em automação industrial, automação residencial e diversas outras áreas. Ele pode operar no modelo **mestre-escravo** (via comunicação serial) ou **cliente-servidor** (quando utilizado em redes *Ethernet*). Nesse modelo, o mestre/cliente envia requisições, e os dispositivos escravos/servidores respondem com dados ou ações.

Segundo o site oficial do *Modbus*, o Modbus Organization (2024), as principais variações do *Modbus* são:

- ***Modbus* ASCII:** A versão original do protocolo, que utiliza codificação em ASCII para transmitir mensagens. Embora mais lenta devido à legibilidade das mensagens, ainda é empregada em sistemas legados ou específicos.
- ***Modbus* RTU (*Remote Terminal Unit*):** É a versão mais amplamente utilizada hoje, pois usa codificação binária e um sistema eficiente de verificação de erros (CRC), permitindo maior densidade de dados e velocidade de transmissão.
- ***Modbus* TCP (*Modbus over Ethernet*):** Implementado sobre a pilha de protocolos TCP/IP, o *Modbus* TCP adiciona um cabeçalho específico (MBAP) às mensagens *Modbus*, tornando-o ideal para redes modernas, com altas taxas de transmissão e maior flexibilidade para integração com sistemas baseados em IP.

Além disso, o *Modbus* é compatível com sua operação em diversos meios físicos, incluindo:

- **RS-232:** Comunicação simples entre dois dispositivos, com alcance de até 30 metros.
- **RS-485:** Suporta até 32 dispositivos em até 1200 metros, sendo muito utilizado em automação industrial.

- ***Ethernet* (*Modbus* TCP):** Comunicação de alta velocidade e integração com sistemas modernos, suportando maiores volumes de dados e redes distribuídas.

Essa flexibilidade permite a utilização do protocolo em diferentes ambientes e requisitos.

2.5. *LoRa e LoRaWAN*

A tecnologia de comunicação sem fio LoRa (*Long Range*) permite a transmissão de dados em longas distâncias, cobrindo amplas áreas, mesmo em ambientes urbanos densos. É ideal para aplicações que transmitem pequenas quantidades de dados com baixa taxa de *bits*, oferecendo um alcance maior do que tecnologias como Wi-Fi, *Bluetooth* ou *Zigbee*, tornando-a adequada para sensores e atuadores de baixo consumo de energia (The Things Network, 2022).

O LoRaWAN (*Long Range Wide Area Network*), por outro lado, é um protocolo de camada MAC (*Media Access Control*) construído sobre a modulação LoRa. É uma camada de software que define como os dispositivos utilizam o *hardware* LoRa, por exemplo, quando transmitem, e o formato das mensagens (The Things Network, 2022). Destaca-se pela comunicação bidirecional e abordagem assíncrona, permitindo maior flexibilidade e eficiência na comunicação. A rede LoRa opera em faixas de frequência não licenciadas (915MHz a 928MHz no Brasil), o que simplifica a implementação, reduz custos e facilita a expansão destas redes, permitindo a conexão de mais dispositivos sem processos complexos de regulamentação.

A arquitetura LoRaWAN é baseada em uma topologia de rede estrela, contendo os seguintes elementos:

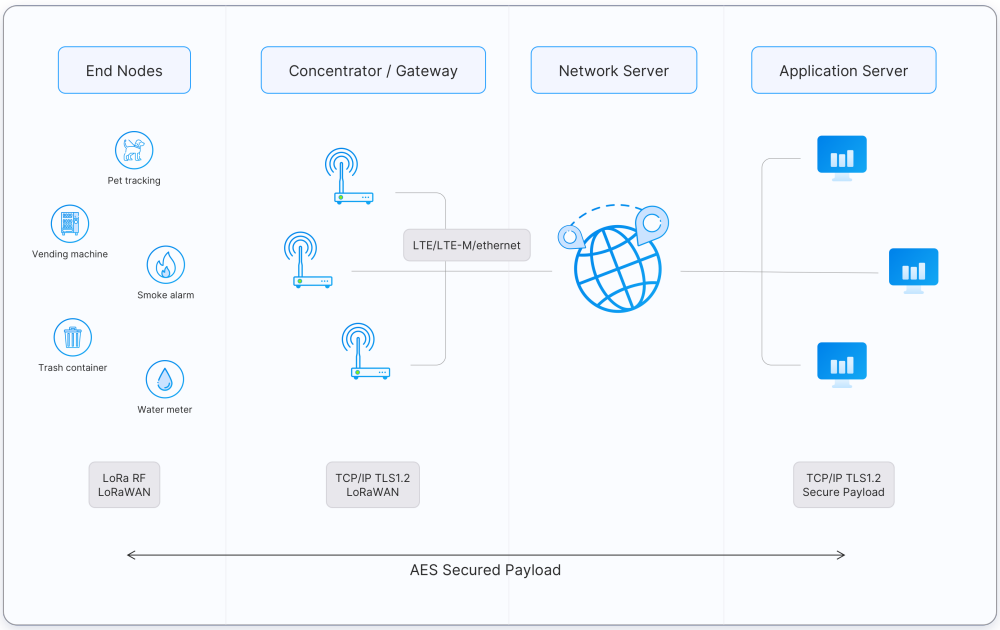

Figura 4. Arquitetura LoRaWAN (The Things Network, 2022).

1. **Dispositivos finais (*end-devices*)**: trocam mensagens em fluxo bidirecional com *gateways* LoRa.

2. ***Gateways***: realizam a ligação entre os *end-devices* e o servidor de rede. Cada *gateway* é registrado em um servidor de rede LoRaWAN. Os *gateways* são conectados ao servidor de rede usando uma infraestrutura de comunicação como links de celular (3G/4G/5G), Wi-Fi ou *Ethernet*.
3. **Servidores de rede (*Network Server*)**: gerencia *gateways*, *end-devices*, aplicativos e usuários em toda a rede LoRaWAN. Têm funções como: estabelecer conexões seguras AES de 128 bits para o transporte de mensagens entre os *end-devices* e o Servidor de Aplicativos (segurança ponta a ponta), validar a autenticidade dos *end-devices* e a integridade das mensagens e selecionar o melhor *gateway* para rotear mensagens de *downlink*.
4. **Servidor de aplicativos (*Application Server*)**: Processa mensagens de dados específicas da aplicação recebidas dos *end-devices*. Ele também gera todas as cargas de *downlink* da camada de aplicação e as envia para os dispositivos finais conectados por meio do servidor de rede. Uma rede LoRaWAN pode ter mais de um servidor de aplicativos. Os dados coletados podem ser interpretados aplicando técnicas como aprendizado de máquina e inteligência artificial para resolver problemas de negócios.
5. **Servidor de ingresso (*Join Server*)**: Auxilia na ativação segura de dispositivos, armazenamento de chave raiz e geração de chave de sessão. O procedimento de adesão é iniciado pelo *end-device*, enviando a mensagem de solicitação de adesão ao servidor de ingresso através do Servidor de Rede. O servidor de ingresso processa a mensagem de solicitação de ingresso, gera chaves de sessão (*NwkSKey* e *AppSKey*) e transfere ambas para o servidor de rede e o servidor de aplicativos, respectivamente.

2.6. MQTT

O MQTT (*Message Queuing Telemetry Transport*) representa um marco na evolução da IoT. Conforme descrito por OASIS (2019), o MQTT é um protocolo de transporte de mensagens cliente-servidor baseado em *publish/subscribe*. É leve, aberto, simples e projetado para ser fácil de implementar. Essas características o tornam ideal para uso em muitas situações, incluindo ambientes restritos, como para comunicação no contexto da IoT, onde é necessária a utilização de código compacto e/ou em situações aonde a largura de banda da rede é limitada.

Entender as vantagens e razões para utilizar o protocolo MQTT é fundamental. Alguns de seus pontos positivos serão explorados com base nos dados fornecidos pela MQTT Organizatation (2022):

- **Leveza e Eficiência**: Projetado para minimizar o uso de recursos pelos dispositivos e a largura de banda da rede, tornando-o ideal para ambientes com recursos limitados.
- **Comunicação Bidirecional**: Facilita a comunicação entre dispositivos e servidores, suportando publicação e assinatura de mensagens, além do envio de mensagens para grupos de dispositivos.
- **Escalabilidade**: Capaz de escalar para suportar milhões de dispositivos em um ecossistema de IIoT (*Industrial IoT*), graças à sua arquitetura distribuída.
- **Níveis de Qualidade de Serviço (QoS)**: Oferece três níveis de QoS para garantir a entrega confiável de mensagens (*At most once*, *At least once* e *Exactly once*),

permitindo aos desenvolvedores escolher o nível de garantia de entrega adequado para suas aplicações.

- **Sessões Persistentes**: Suporta sessões persistentes entre dispositivos e servidores, garantindo a continuidade da comunicação mesmo em caso de desconexão temporária da rede.
- **Recursos de Segurança**: Dispõe de recursos robustos de segurança, incluindo criptografia TLS para confidencialidade das mensagens e protocolos de autenticação para verificação de clientes.

O protocolo opera com um modelo de comunicação *publish/subscribe*. Neste modelo, existem três componentes principais (Figura 5): o *Publisher* (publicador), o *Broker* e o *Subscriber* (assinante).

No processo de comunicação, o *Publisher* é responsável por enviar mensagens para o *Broker*. O *Broker* atua como um intermediário que recebe, armazena e encaminha as mensagens recebidas pelo *Publisher* para os *Subscribers* interessados. Os *Subscribers* se inscrevem em tópicos específicos de interesse e recebem as mensagens publicadas pelo *Publisher* através do *Broker*, garantindo uma comunicação direcionada e eficiente entre os dispositivos.

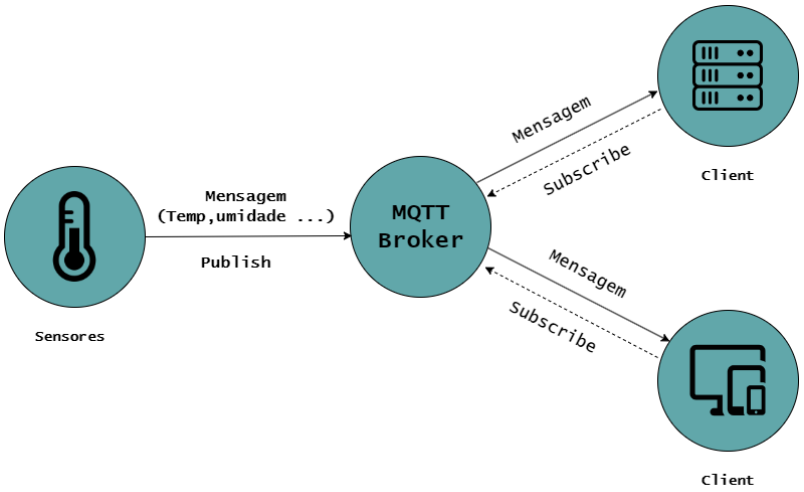

Figura 5. Comunicação MQTT, adaptada de MQTT Organizatation (2022).

Em conclusão, o MQTT desempenha um papel fundamental na construção de sistemas de comunicação robustos e confiáveis em ambientes e IoT. Sua leveza, eficiência, escalabilidade, confiabilidade e recursos de segurança fazem dele a escolha ideal para conectar dispositivos e trocar dados pela Internet.

2.7. *ThingsBoard*

O *ThingsBoard* é uma plataforma de código aberto voltada para a gestão da infraestrutura IoT, facilitando assim o desenvolvimento, gerenciamento e escalabilidade de projetos de IoT. Conforme descrito em sua documentação oficial (Thingsboard, 2024), a plataforma oferece uma solução pronta para uso, seja na nuvem ou localmente, proporcionando infraestrutura robusta para aplicações IoT.

Entre suas principais funcionalidades, o *ThingsBoard* permite a coleta, visualização e análise de dados de dispositivos conectados, além de suportar a criação de *dashboards* dinâmicos e interativos. Ele é compatível com diferentes protocolos de comunicação, como MQTT e HTTP, o que garante flexibilidade na integração de dispositivos e sistemas. A plataforma possibilita não apenas a transmissão de dados, mas também o envio de comandos para dispositivos, a configuração remota de parâmetros e a definição de alertas baseados em eventos complexos. Essas funcionalidades tornam o *ThingsBoard* uma solução versátil para aplicações em diversos contextos IoT, como monitoramento ambiental, controle industrial e gestão de energia.

2.8. Trabalhos Correlatos

É fundamental situar o presente trabalho no contexto das pesquisas relacionadas já realizadas. Nesta seção, serão apresentados alguns estudos e projetos relevantes que abordam temas similares ou complementares ao proposto neste trabalho. Para a identificação desses estudos, foram realizadas buscas em bases de dados de alto impacto, como o *IEEE Xplore*, acessada por meio do portal de periódicos da Capes. As palavras-chave utilizadas incluíram termos relacionados a comunicação em estações meteorológicas, 4G, IoT, microcontroladores e *dataloggers*, e os critérios de seleção consideraram a relevância do tema, a atualidade das publicações e a aplicação prática dos resultados apresentados. Além disso, também foi realizada uma análise no repositório de Trabalhos de Conclusão de Curso (TCCs) do IFSC-Lages. Essa consulta possibilitou identificar abordagens locais e práticas que compartilham contextos semelhantes.

Um trabalho correlato relevante é o *Development of an IoT communication module for energy sharing communities management and monitorization* (Faria e et al., 2022). Enquanto o presente trabalho concentra-se na proposta de um módulo de comunicação para as estações meteorológicas da EPAGRI, o trabalho correlato aborda a facilitação do gerenciamento e monitoramento de comunidades de compartilhamento de energia. Ambos os projetos compartilham a utilização de tecnologias LPWAN, enfatizando a importância dessas tecnologias emergentes na facilitação da comunicação em aplicações IoT em larga escala e em diferentes contextos. Portanto, considerando a similaridade na abordagem tecnológica e o objetivo compartilhado de desenvolver módulos de comunicação para aplicações específicas, o trabalho correlato oferece *insights* valiosos sobre o uso e a relevância das tecnologias LPWAN em diferentes domínios. Esses *insights* serão aproveitados para refinar a abordagem adotada neste trabalho e garantir a integração eficiente das tecnologias LPWAN no módulo de comunicação proposto.

Outro trabalho correlato relevante é o *Application of 4G IoT Card in Automatic Weather Station* (Sun e Wu, 2020). Este estudo explora a aplicação de um cartão IoT 4G em estações meteorológicas automáticas, visando melhorar a comunicação e o monitoramento nessas estações. O uso do cartão IoT 4G permite uma conectividade mais rápida e estável, possibilitando a transmissão de dados meteorológicos em tempo real para os centros de monitoramento. Além disso, o estudo investiga os benefícios adicionais proporcionados pela tecnologia 4G, como maior largura de banda e maior alcance de cobertura, que podem aprimorar significativamente a capacidade de coleta e análise de dados meteorológicos. Este também oferece uma visão aprofundada sobre os desafios de segurança do uso de cartões SIM de redes 2G, fornecendo uma perspectiva relevante para a evolução das tecnologias de comunicação em sistemas meteorológicos automáticos.

No estudo *"Wireless IoT communication module with low power consumption for a soil moisture and salinity sensor"* (Łostowski et al., 2020) foi proposto a criação de um módulo de comunicação sem fio baseado em tecnologia LPWAN para troca de dados com um banco de dados externo. O objetivo principal é melhorar a eficiência da gestão agrícola ao permitir a coleta de dados em tempo real sobre a umidade do solo e a salinidade, essenciais para a tomada de decisões informadas pelos agricultores. O projeto destaca a importância de soluções de comunicação robustas e de baixo consumo de energia para garantir a conectividade em ambientes remotos, onde a infraestrutura de rede tradicional pode ser limitada. Além disso, enfatiza a necessidade de adaptar os dispositivos IoT às condições específicas do ambiente agrícola, como a duração da bateria e a transmissão de grandes volumes de dados de forma eficiente. Essa pesquisa contribui significativamente para o avanço da agricultura de precisão, fornecendo ferramentas essenciais para o monitoramento e aprimoramento das práticas agrícolas.

3. Modelagem

Nesta seção, será definida a modelagem da solução proposta. A seção abordará a estrutura e os componentes essenciais do módulo de comunicação, com base nos conhecimentos descritos na seção 2. Conforme ilustrado na Figura 6, será apresentado um diagrama estático que oferece uma visão geral da solução, dando uma ideia da topologia da mesma. Esse diagrama servirá como uma base visual para entender a interação entre os elementos do sistema e a forma como eles contribuem para o funcionamento global da solução.

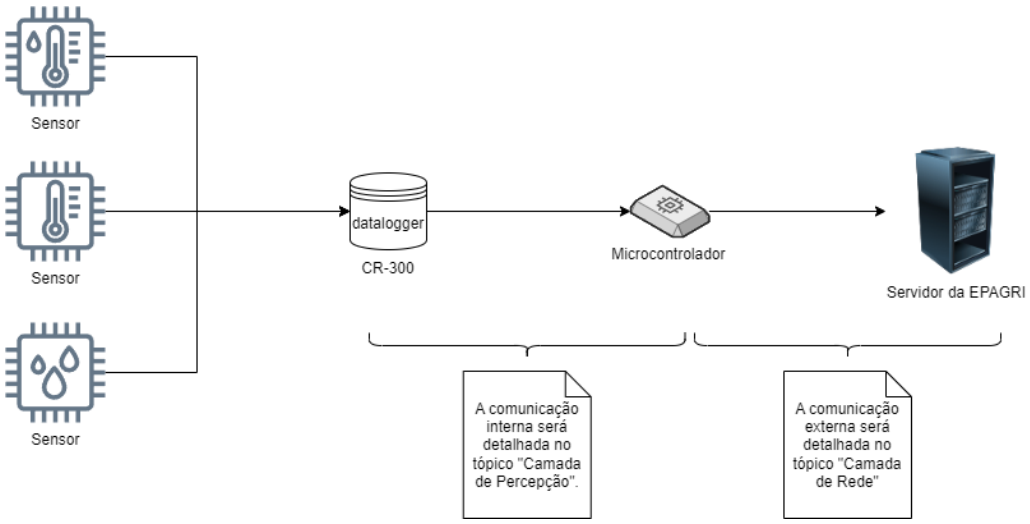

Figura 6. Topologia da solução.

Conforme descrito na Seção 2.2, a arquitetura da IoT é dividida em cinco camadas. Neste caso, foi necessário atuar nas camadas de percepção, rede e *middleware*, aplicação e negócios.

3.1. Camada de Percepção

Na camada de percepção, que representa a interface direta e física entre o ambiente real e o ecossistema da IoT, a principal tarefa foi a construção do *firmware* responsável pela

extração dos dados que o *datalogger* recebe diretamente dos sensores e armazena. Além disso, o *firmware* prepara esses dados para serem encaminhados à camada de rede.

A extração dos dados do *datalogger* foi realizada por meio do protocolo *Modbus*, escolhido por sua confiabilidade e ampla compatibilidade com o *datalogger* CR300 e o microcontrolador ESP32. Nesse processo, o *datalogger* atua como *slave*, respondendo às requisições (*requests*) enviadas pelo microcontrolador, que desempenha o papel de *master*.

Na Figura 7, é apresentado um diagrama de atividade que ilustra o funcionamento da camada de percepção.

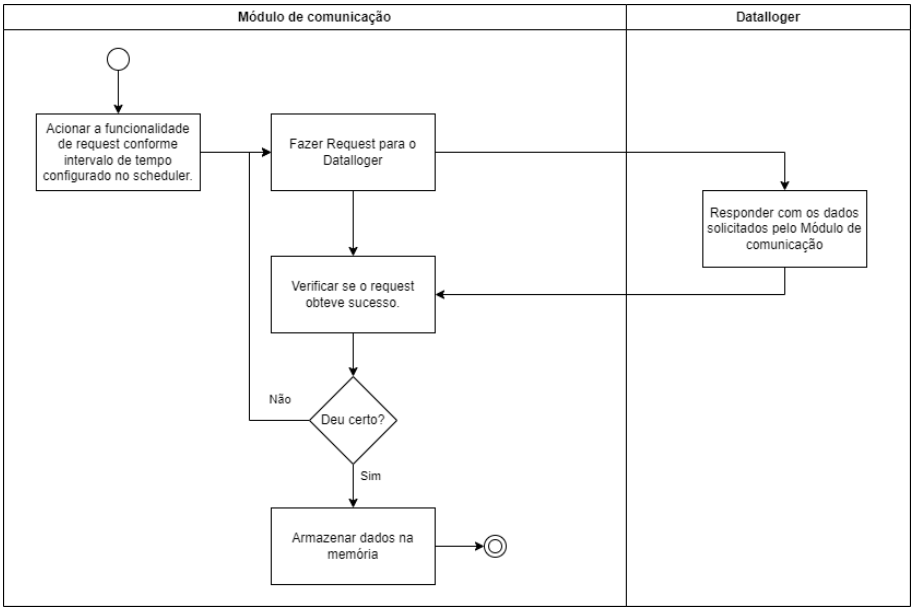

Figura 7. Diagrama de atividades da comunicação interna.

3.2. Camada de Rede

A camada de rede é responsável por conectar os dispositivos IoT à infraestrutura de rede e sistemas, facilitando a comunicação dos dados provenientes da camada de percepção para outras partes interessadas. O principal objetivo desta camada é externalizar os dados, permitindo que sejam processados, analisados e utilizados de forma eficiente.

Para alcançar esse objetivo, foram empregadas tecnologias como 4G, Wi-Fi, e MQTT, que garantem uma comunicação robusta e confiável. Além disso, a camada de rede desempenha um papel crucial na viabilização da atualização remota do *firmware*, permitindo a comunicação bidirecional entre o sistema e os dispositivos, o que possibilita tanto o envio quanto o recebimento de dados.

A comunicação via MQTT, que pode operar sobre tecnologias como 4G ou Wi-Fi, permite tanto o envio quanto o recebimento de dados. Na Figura 8, é apresentado um diagrama de atividade que ilustra o processo de envio de dados via MQTT.

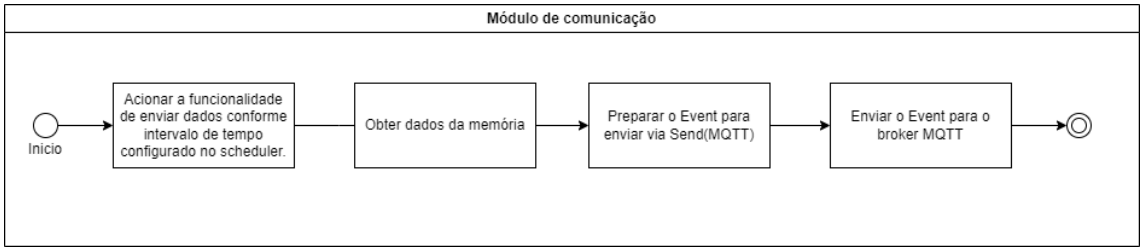

Figura 8. Diagrama de atividades para o envio de dados via MQTT (4G ou Wi-Fi).

Por conta dos desafios encontrados na comunicação entre o dispositivo implementado (ATL-100) e o *datalogger* da EPAGRI (CR-300), não houve tempo hábil para o desenvolvimento e teste do modelo de comunicação LoRa/LoRaWAN. No entanto, o diagrama de atividade que ilustra o processo de envio de dados utilizando este protocolo é apresentado na Figura 9.

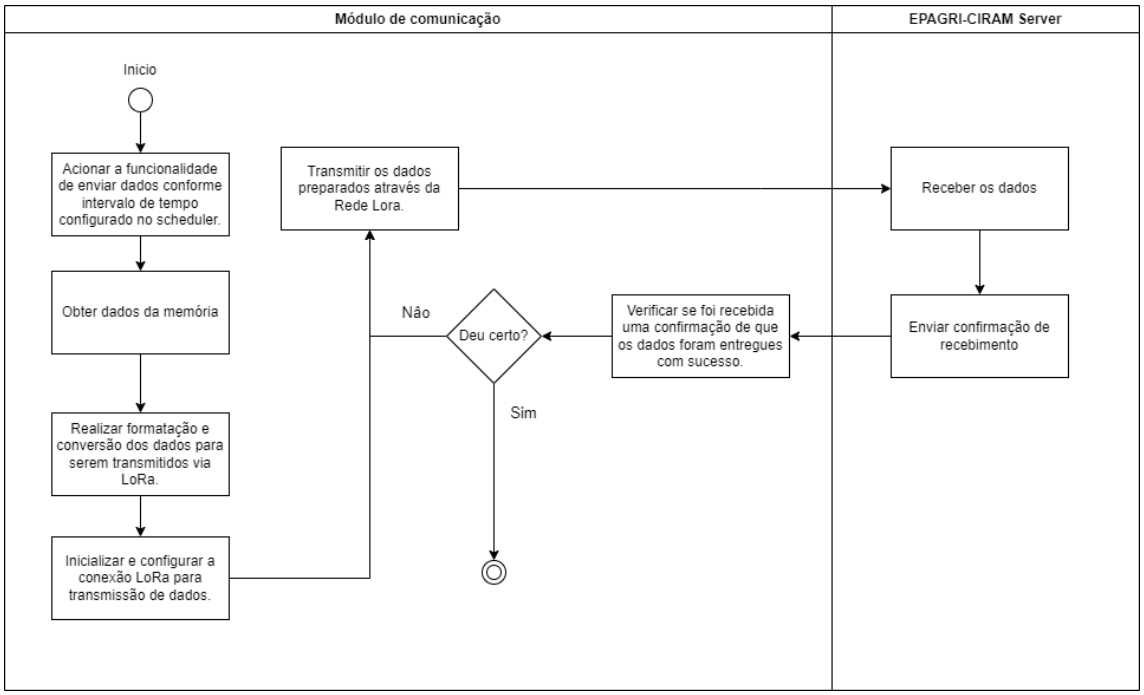

Figura 9. Diagrama de atividades para o envio de dados via LoRa/LoRaWAN.

3.3. Camada de *Middleware*

A camada de *middleware* desempenha um papel crucial ao facilitar a comunicação entre os dispositivos finais e os servidores, garantindo que os dados sejam processados e armazenados de maneira correta. No contexto desta solução, o *middleware* é implementado utilizando a plataforma *ThingsBoard*, que atua como intermediário ao receber os dados enviados diretamente pelo MCU, processá-los e armazená-los em seu banco de dados.

O *ThingsBoard* utiliza uma arquitetura baseada em mensagens que emprega, dentre outros, o MQTT ou o HTTP como protocolos para a comunicação com os dispositivos.

Ele é responsável por normalizar os dados recebidos, gerenciar dispositivos conectados, e fornecer ferramentas para visualização e análise em tempo real, bem como para armazenamento histórico. Essa flexibilidade permite que o *ThingsBoard* não apenas processe os dados, mas também os integre a outras plataformas ou sistemas, caso necessário.

Dessa forma, o *middleware* do *ThingsBoard* assegura a confiabilidade e a escalabilidade da solução proposta, atendendo aos requisitos de processamento e armazenamento de dados deste trabalho.

3.4. Camada de Aplicação e Negócios

As camadas de aplicação e negócios foram planejadas para ser desenvolvidas utilizando a ferramenta *ThingsBoard*. Essa plataforma permite simular o armazenamento, o processamento e a visualização de dados, oferecendo uma visão preliminar de como o sistema funcionaria de forma integrada. Dessa maneira, seria possível representar como a EPAGRI/CIRAM utilizaria a aplicação. Contudo, essa implementação específica é considerada um recurso adicional, pois, apesar de sua importância, não constitui o escopo deste trabalho. A implementação final ficará sob responsabilidade da EPAGRI/CIRAM, que determinará a melhor forma de consumo dos dados de acordo com suas necessidades. Neste trabalho, o escopo é demonstrar uma solução similar àquela atualmente utilizada no portal *Agroconnect*[2].

4. Desenvolvimento

Nesta seção, serão abordados os detalhes de implementação de cada camada da arquitetura IoT da solução proposta.

4.1. Camada de Percepção

Durante o desenvolvimento da camada de percepção atuamos em 2 pontos: validação da hipótese de coleta de dados do *datalogger* CR-300 e elaboração do projeto da PCB a ser utilizada na aplicação.

Inicialmente, construímos uma prova de conceito (PoC - *Proof of Concept*) para validar o método de extração de dados do *datalogger* CR-300, responsável por armazenar as informações dos sensores. Essa PoC foi desenvolvida em *CSharp Windows Form* e utilizou o protocolo *Ethernet over* USB para estabelecer a comunicação com o *datalogger*, possibilitando a realização de chamadas HTTP entre o computador e o *datalogger*.

A PoC realiza chamadas HTTP do tipo GET para a API do *datalogger*, permitindo que o usuário configure a requisição diretamente na interface da aplicação, conforme apresentado na Figura 10.

[2] `https://ciram.epagri.sc.gov.br/agroconnect/`

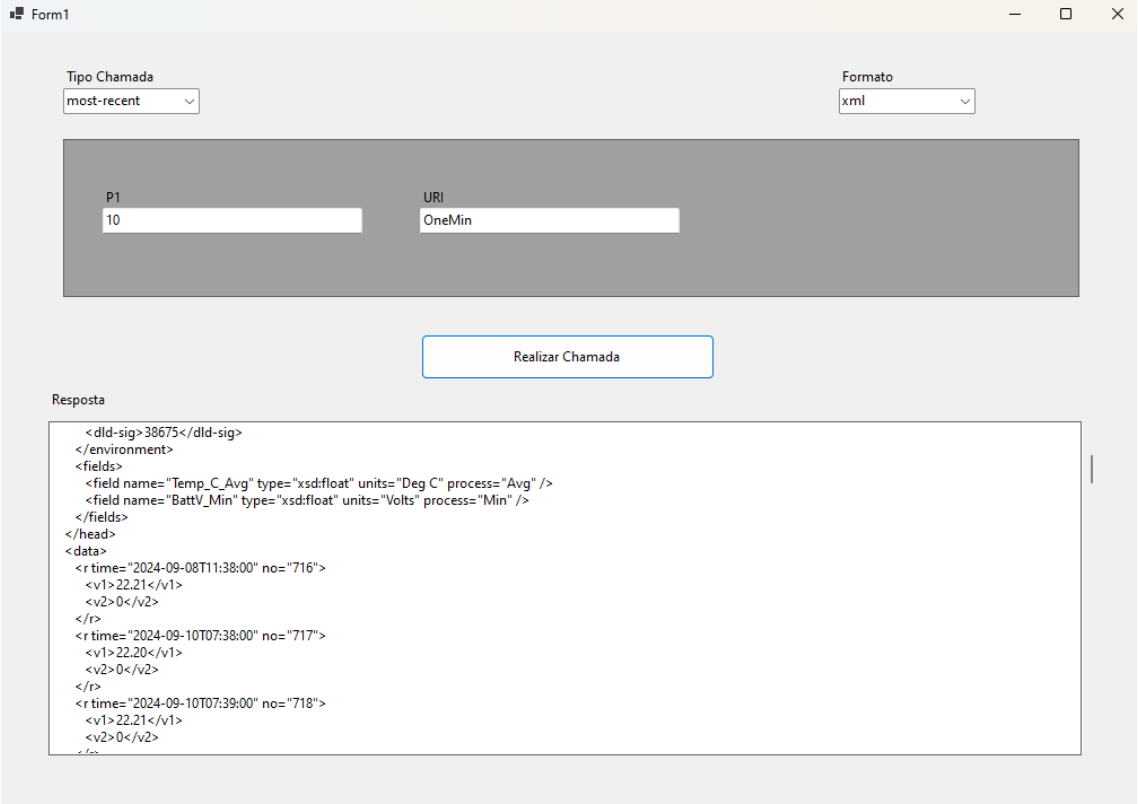

Figura 10. PoC de comunicação com o CR-300.

Por exemplo, a chamada apresentada na Figura 10 irá executar a seguinte chamada HTTP apresentada na Figura 11.

```
http://192.168.66.1/?command=DataQuery&mode=most-recent&format=xml&uri=dl:OneMin&p1=10
```

Figura 11. Exemplo de chamada *most-recent*.

Nessa URL, 192.168.66.1 é o endereço padrão do CR-300, enquanto os parâmetros configurados na tela da PoC especificam que a chamada buscará as 10 atualizações mais recentes da telemetria no formato XML, utilizando o modo "*most-recent*". No exemplo da imagem 11, *"OneMin"* é o nome da tabela de onde os dados são extraídos. O principal objetivo da PoC foi explorar os diferentes tipos de chamadas disponíveis na API do CR-300, permitindo testar e configurar consultas tanto para dados recentes quanto para intervalos de datas.

Finalmente, desenvolvemos o projeto de *hardware*, incluindo a placa de circuito impresso (PCB), sobre o qual a aplicação será executada. Na imagem 12 está a modelagem 3D de como ficou o projeto final.

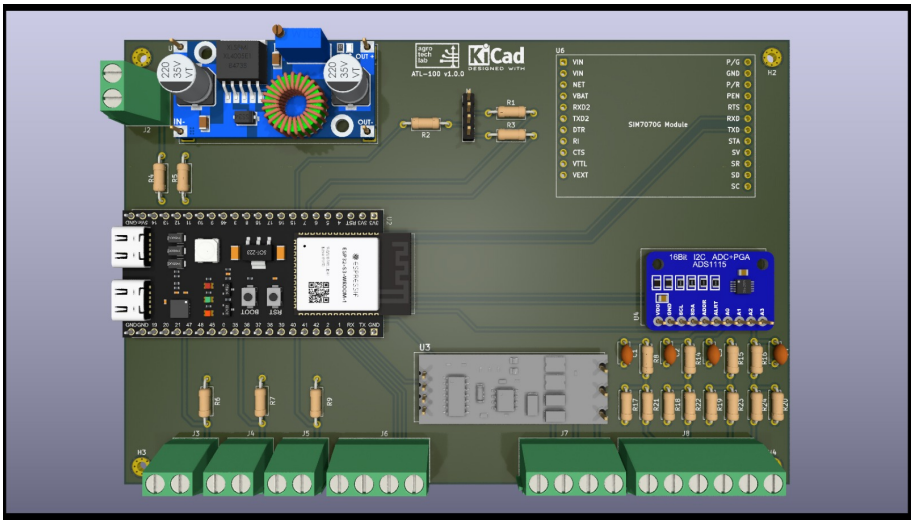

Figura 12. Visão geral da PCB desenvolvida.

Inicialmente, a comunicação entre o microcontrolador e o *datalogger* CR-300 foi projetada para ser realizada por meio de chamadas HTTP utilizando o protocolo *Ethernet over* USB. Essa abordagem demonstrou-se viável durante a etapa de prova de conceito (PoC), onde foi validado o método de extração de dados através de chamadas HTTP realizadas a partir de um computador. No entanto, na implementação prática com o microcontrolador, surgiram desafios significativos relacionados à utilização do protocolo RNDIS (*Remote Network Driver Interface Specification*), necessário para estabelecer a comunicação entre o CR-300 e o ATL-100. Enquanto o RNDIS é suportado nativamente em sistemas operacionais *Windows*, sua implementação em microcontroladores revelou-se complexa e inviável devido a limitações de tempo e recursos de desenvolvimento.

Diante dessas dificuldades, decidiu-se adotar o protocolo *Modbus*, que, além de ser amplamente utilizado em aplicações industriais, ofereceu uma solução mais prática e eficiente para o contexto do projeto. O *Modbus* permitiu estabelecer a comunicação direta entre o microcontrolador e o *datalogger*, simplificando a arquitetura e eliminando a dependência de protocolos mais complexos, como o RNDIS.

Tendo em vista a mudança de protocolo de comunicação, foi preciso escrever um programa para o *datalogger* para adequar a integração com o microcontrolador. O programa em questão foi escrito em CRBasic, a linguagem de programação proprietária da *Campbell Scientific* (Figura 13) e carregado no *datalogger* CR-300 para habilitar a comunicação via protocolo *Modbus* utilizando a interface RS-232 do mesmo.

O programa realiza as seguintes funções principais:

- **Coleta de Dados**: O código coleta periodicamente os dados de tensão do sistema de alimentação, temperatura do equipamento (CR-300) e dados de temperatura e umidade do ar, a partir do sensor *dualBASE TUSensDB* com conexão SDI-12.
- **Configuração *Modbus***: O *datalogger* é configurado como um dispositivo *Modbus Slave*, permitindo que o microcontrolador se conecte a ele e acesse os dados via requisições *Modbus*.
- **Tabela de Dados**: Uma tabela de dados é criada no programa para armazenar e organizar as informações coletadas antes de serem transmitidas.

```
'Declare Public Variables
Public BattV
Public PTemp
Public TRHData(2)
Public Modbus(4)
Public ModbusCoil(1) As Boolean

'Define aliases
Alias TRHData(1) = AirTemp
Alias TRHData(2) = AirHumid

'Define values units
Units BattV = Volts
Units PTemp = Deg C
Units AirTemp = Deg C
Units AirHumid = %

'Define Data Tables.
DataTable (Test,1,-1) 'Set table size to # of records, or -1 to autoallocate.
  DataInterval (0,15,Sec,10)
  Minimum (1,BattV,FP2,False,False)
  Sample (1,PTemp,FP2)
EndTable

'Main Program
BeginProg
  'Use SerialOpen to set RS232 options for Modbus Slave Instruction
  SerialOpen(ComRS232,115200,3,0,1000)
  'Modbus Slave Instruction
  ModbusServer(ComRS232,115200,1,Modbus(),ModbusCoil(),2)
  'Main Scan
  Scan (1,Sec,0,0)
    'Get battery voltage
    Battery(BattV)
    'Get panel temperature
    PanelTemp(PTemp,60)
    'Get air temperature and humidity from dualBASE SDI12 sensor
    SDI12Recorder(TRHData(),1,"0","M!",1,0)
    'Call Output Tables
    CallTable Test
    'Copy values/measuremenets to Modbus Array
    Modbus(1) = BattV
    Modbus(2) = PTemp
    Modbus(3) = AirTemp
    Modbus(4) = AirHumid
  NextScan
EndProg
```

Figura 13. Programa feito para o *datalogger* considerando o protocolo *Modbus*.

Além da configuração do *datalogger*, também foi necessário desenvolver o *firmware* do microcontrolador, com a implementação do protocolo *Modbus*. Este *firmware*, baseado no *framework* ESP-IDF versão 5.3.1, adota o *FreeRTOS* como sistema operacional de tempo real para sistemas embarcados. Na Figura 14 é apresentado o código do *firmware* responsável pela inicialização do Mestre *Modbus* na interface RS-232. Além disso, o código configura a interface serial no modo *half-duplex* e inicializa o descritor *Modbus*, garantindo uma comunicação eficiente entre os dispositivos.

O descritor *Modbus*, por sua vez, é responsável por mapear o número do registrador de dados que se deseja consultar, bem como definir parâmetros como a quantidade de dados, seu tamanho, formato e o endereço do escravo *Modbus*. Esses aspectos são essenciais para garantir que o fluxo de informações ocorra de maneira eficiente e estruturada. A implementação dessa funcionalidade específica é apresentada na Figura 15, detalhando como os parâmetros são configurados no *firmware*.

```
    /* Set Modbus UART pin */
    err = uart_set_pin(ATL_MB_RS232_PORT_NUM, CONFIG_ATL_RS232_TXD_GPIO, CONFIG_ATL_RS232_RXD_GPIO, UART_PIN_NO_CHANGE, UART_PIN_NO_CHANGE);
    if (err != ESP_OK) {
        ESP_LOGE(TAG, "Modbus Master UART set pin fail!");
        goto error_proc;
    }

    /* Start Modbus Master controller */
    err = mbc_master_start();
    if (err != ESP_OK) {
        ESP_LOGE(TAG, "Modbus Master controller start fail!");
        goto error_proc;
    }

    /* Set driver mode to Half Duplex */
    err = uart_set_mode(ATL_MB_RS232_PORT_NUM, UART_MODE_UART);
    if (err != ESP_OK) {
        ESP_LOGE(TAG, "Modbus Master UART set mode fail!");
        goto error_proc;
    }

    /* Set Modbus Master stack descriptor */
    vTaskDelay(5);
    err = mbc_master_set_descriptor(&atl_modbus_device_parameters[0], atl_modbus_get_num_device_params());
    if (err != ESP_OK) {
        ESP_LOGE(TAG, "Modbus Master set descriptor fail!");
        goto error_proc;
    }

    ESP_LOGI(TAG, "Modbus Master initialized at RS-232 interface!");
    modbus_master_rs232_initialized = true;
    return err;

    /* Error procedure */
error_proc:
    atl_led_builtin_blink(6, 100, 255, 0, 0);
    ESP_LOGE(TAG, "Error: %d = %s", err, esp_err_to_name(err));
    return err;
}
```

Figura 14. Inicialização do Mestre *Modbus* no ATL-100.

```
/**
 * @brief Modbus descriptor
 * @details CID - field in the table (must be unique).
 *          Param Name - Name of the parameter.
 *          Units - Units of the parameter.
 *          Modbus Slave Addr - field defines slave address of the device with correspond parameter.
 *          Modbus Reg Type - Type of Modbus register area (Holding register, Input Register and such).
 *          Reg Start - field defines the start Modbus register number.
 *          Reg Size - defines the number of registers for the characteristic accordingly.
 *          Instance Offset - defines offset in the appropriate parameter structure that will be used as instance to save parameter value.
 *          Data Type - specify value type of CID.
 *          Data Size - specify value size of CID.
 *          Parameter Options - field specifies the options that can be used to process parameter value (limits or masks).
 *          Access Mode - can be used to implement custom options for processing of characteristic (Read/Write restrictions, factory mode values and etc).
 */
const mb_parameter_descriptor_t atl_modbus_device_parameters[] = {
    { CID_HOLD_BATT_VOLTAGE, ATL_MB_STR("CR300_BattV"), ATL_MB_STR("V"), 2, MB_PARAM_HOLDING, 40001, 2,
            HOLD_OFFSET(holding_data9), PARAM_TYPE_FLOAT, 2, ATL_MB_OPTS( 0, 40, 1 ), PAR_PERMS_READ },
    { CID_HOLD_PTEMP, ATL_MB_STR("CR300_PTemp"), ATL_MB_STR("oC"), 2, MB_PARAM_HOLDING, 2, 2,
            HOLD_OFFSET(holding_data10), PARAM_TYPE_FLOAT, 2, ATL_MB_OPTS( -40, 80, 0.1 ), PAR_PERMS_READ },
    { CID_HOLD_TEMPERATURE, ATL_MB_STR("CR300_AirTemp"), ATL_MB_STR("oC"), 2, MB_PARAM_HOLDING, 4, 2,
            HOLD_OFFSET(holding_data11), PARAM_TYPE_FLOAT, 2, ATL_MB_OPTS( -40, 80, 0.1 ), PAR_PERMS_READ },
    { CID_HOLD_HUMIDITY, ATL_MB_STR("CR300_AirHumid"), ATL_MB_STR("%RH"), 2, MB_PARAM_HOLDING, 6, 2,
            HOLD_OFFSET(holding_data12), PARAM_TYPE_FLOAT, 2, ATL_MB_OPTS( 0, 100, 0.1 ), PAR_PERMS_READ },
};
```

Figura 15. Descritor do Mestre *Modbus* no ATL-100.

A Figura 16 apresenta a estação montada, que integra os componentes do sistema, incluindo sensores, *datalogger*, módulo de comunicação e fontes de alimentação. Essa configuração reflete um cenário de aplicação real, simulando condições de operação em campo para validar o funcionamento do programa desenvolvido. O objetivo é garantir a coleta eficiente dos dados e a comunicação confiável entre o *datalogger* e o microcontrolador utilizando o protocolo *Modbus*.

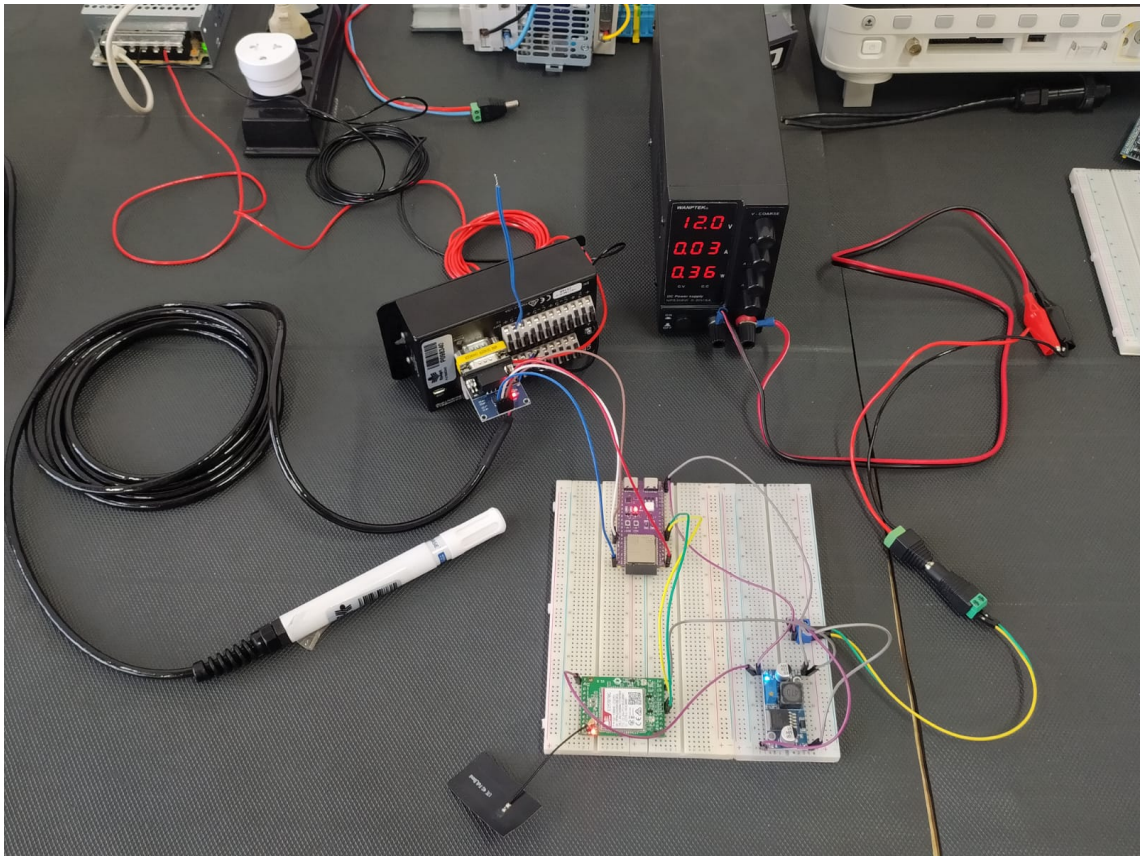

Figura 16. Protótipo do ATL-100 conectado ao CR-300 com o sensor TUSensDB.

4.2. Camada de *Middleware*

Nesta camada, o foco principal foi compreender e utilizar as funcionalidades do *middleware* oferecido pelo *ThingsBoard*. A plataforma foi utilizada para gerenciar a coleta, o processamento e o armazenamento dos dados em seus servidores de forma eficiente.

O *ThingsBoard* permite a configuração de fluxos de dados personalizados, viabilizando o armazenamento centralizado e facilitando a visualização e análise dos dados de telemetria. Além disso, a plataforma suporta a reparametrização dos dispositivos por meio de comandos enviados remotamente, permitindo ajustes como tempos de amostragem e outras variáveis operacionais.

Essa abordagem eliminou a necessidade de uma implementação própria de *middleware*, aproveitando os recursos robustos e integrados do *ThingsBoard* para integrar e gerenciar os dados coletados, atendendo aos requisitos do sistema.

As únicas customizações realizadas nesta camada ocorreram dentro do próprio *ThingsBoard*, onde foram implementadas regras específicas para processar os dados recebidos do MCU. Essas regras, baseadas nos valores enviados, geram alertas automaticamente, contribuindo para a automação e a eficiência do sistema. Como exemplo, foram criadas regras de alerta para níveis de umidade e temperatura, ilustradas nas Figuras 17 e 18, respectivamente.

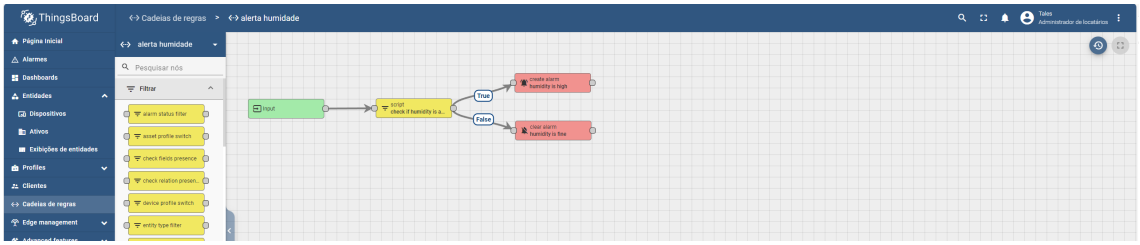

Figura 17. Configuração do alerta de umidade.

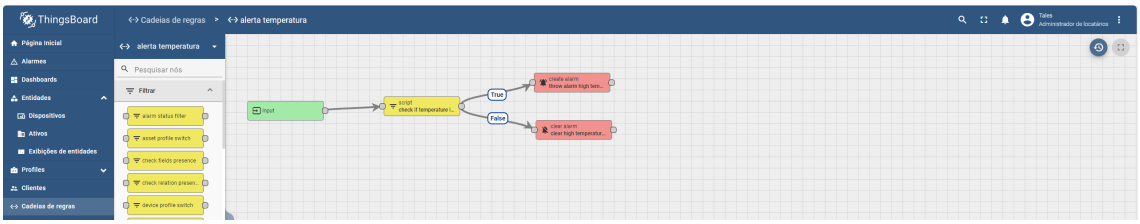

Figura 18. Configuração do alerta de temperatura.

4.3. Camada de Rede

Na camada de rede, foi desenvolvida uma infraestrutura robusta que permite ao MCU se comunicar via 4G e Wi-Fi, possibilitando a transmissão de dados para o servidor da plataforma *ThingsBoard* e, futuramente, para os servidores da EPAGRI/CIRAM. Para viabilizar essa comunicação nos protocolos 4G e Wi-Fi, implementamos o protocolo MQTT utilizando a biblioteca ESP-MQTT, o que possibilita uma comunicação bidirecional entre o MCU e o servidor do *ThingsBoard*. Com isso, não apenas o envio de dados é possível, mas também a recepção de comandos, permitindo a reparametrização do *firmware* conforme necessário.

Antes de implementar o MQTT diretamente no MCU, realizamos testes preliminares de comunicação e validação do envio de dados para o *ThingsBoard* por meio do *Wokwi*[3]. Essa abordagem permitiu avaliar o funcionamento da comunicação em um ambiente controlado e identificar eventuais ajustes necessários.

A integração completa do sistema ao servidor *ThingsBoard* foi concluída com sucesso, e mais detalhes sobre esse processo serão discutidos na seção referente à camada de aplicação e negócios.

4.4. Camada de Aplicação e Negócios

Na camada de aplicação e negócios, potencializamos o valor dos dados ao transformá-los em gráficos, proporcionando maior visibilidade por meio do *ThingsBoard*[4]. Criamos gráficos que se aproximam dos utilizados pela EPAGRI/CIRAM, permitindo a visualização de múltiplas estações em um único painel (Figura 19).

[3]`https://wokwi.com/projects/413725249610130433`
[4]`https://agrotechlab.lages.ifsc.edu.br:8080`

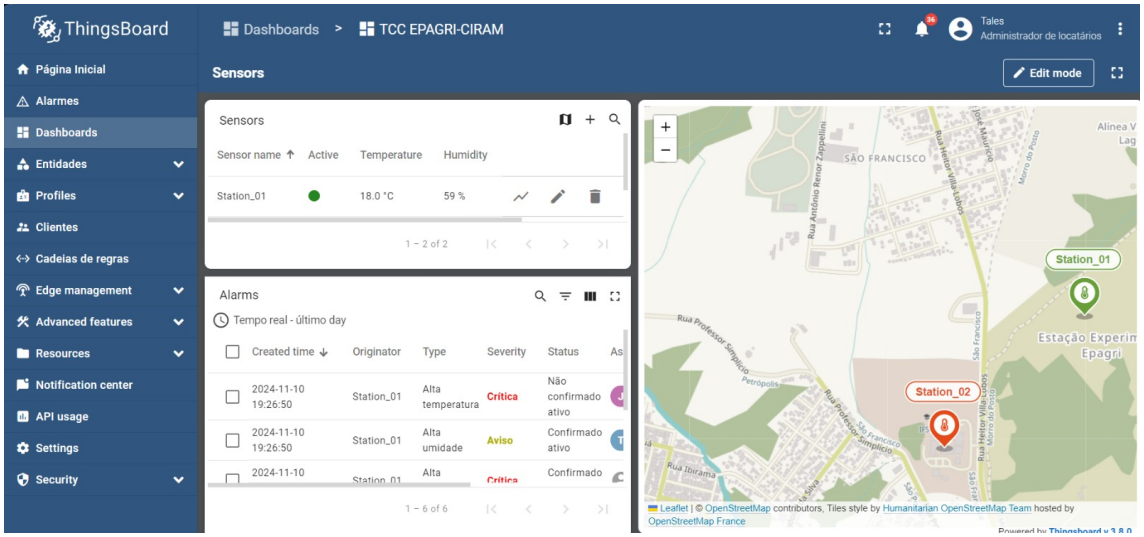

Figura 19. *Dashboard* principal.

Além disso, estruturamos toda a navegação do sistema, o que possibilita tanto uma visão geral das estações quanto um detalhamento específico de cada uma (Figura 20). Dessa forma, conseguimos simular de maneira mais realista o uso do sistema pelos usuários finais.

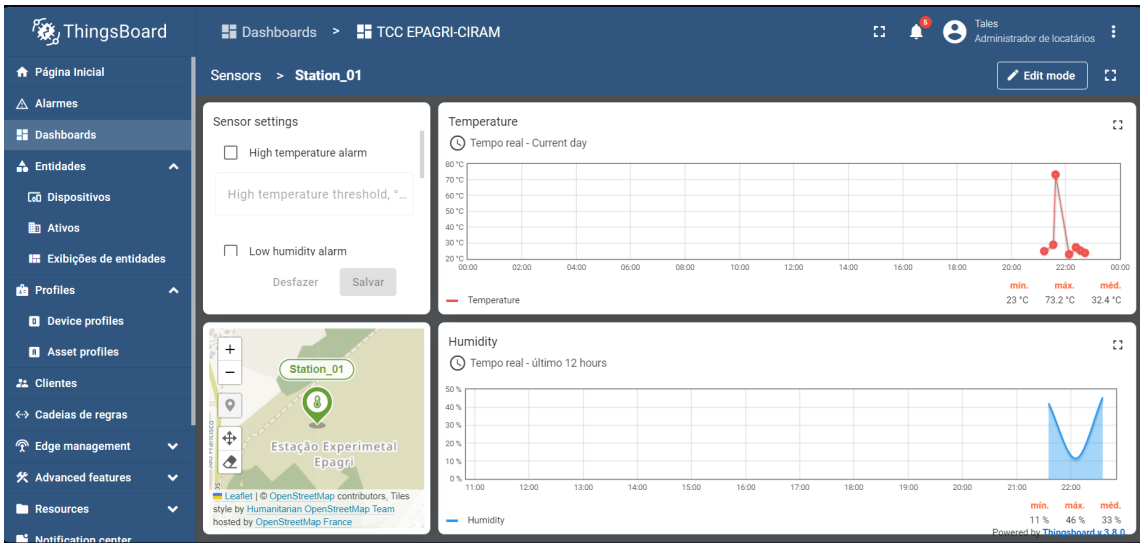

Figura 20. *Dashboard* específico de cada estação.

O *ThingsBoard* também permite a reparametrização dos atributos do *firmware*, completando o fluxo desejado: tanto o envio de dados para o servidor quanto a reconfiguração do *firmware* por meio de comandos enviados pela plataforma. Além disso, implementamos a configuração de alarmes, como limites de temperatura e umidade, que são disparados quando os sensores ultrapassam valores pré-definidos, conforme ilustrado na Figura 21. Esses alarmes podem ser atribuídos a usuários específicos, permitindo que

sejam notificadas e tomadas as ações cabíveis de forma ágil e eficiente, melhorando a gestão dos incidentes.

Figura 21. Página de alarmes *ThingsBoard*.

5. Testes e Resultados

Os testes realizados neste trabalho tiveram como objetivo validar o funcionamento das soluções propostas, incluindo comunicação, processamento de dados e integração com plataformas de monitoramento. Os resultados obtidos são apresentados a seguir.

5.1. Coleta de Dados

A comunicação foi testada utilizando o *datalogger* CR-300 conectado ao sensor de temperatura e umidade do ar TUSensDB (SDI12). Os dados coletados foram processados e transmitidos através de duas abordagens principais:

- ***Ethernet Over* USB - API REST:** Inicialmente, considerou-se realizar a comunicação interna via API REST. No entanto, essa abordagem mostrou-se inviável devido à ausência de bibliotecas estáveis que implementassem o protocolo RNDIS no microcontrolador ESP32. Por esse motivo, a ideia foi abandonada, permanecendo como uma possibilidade futura caso uma biblioteca estável seja encontrada ou desenvolvida.
- **Protocolo *Modbus*:** Diante da inviabilidade do RNDIS, optou-se por implementar o *Modbus* para comunicação direta entre o *datalogger* e o microcontrolador ESP32. Os testes demonstraram leituras consistentes, confirmando a eficácia dessa abordagem.

Com base nos testes realizados, conclui-se que a funcionalidade de comunicação e coleta de dados proposta foi implementada com sucesso, atendendo plenamente aos requisitos definidos para o sistema.

5.2. Envio de Dados

A comunicação e o envio de dados foram testados utilizando o protocolo MQTT (4G e Wi-Fi), alcançando sucesso nos testes realizados. Com isso, o fluxo básico e essencial para a conclusão deste trabalho foi estabelecido, garantindo a operação completa: a extração dos dados via *Modbus* e o envio desses dados por meio do protocolo MQTT para consumo por outras aplicações.

5.3. Reparametrização

Os testes de reparametrização foram realizados utilizando o *ThingsBoard* e o microcontrolador ESP32 (MCU-ESP32), que recebia as mensagens provenientes do *ThingsBoard*. Essa funcionalidade foi implementada parcialmente: atualmente, é possível reparametrizar variáveis enviadas pelo *ThingsBoard*, como tempo de amostragem, coordenadas, entre outras.

Apesar da reparametrização do *firmware* do *datalogger* não ter sido implementada, a funcionalidade de reparametrização do MCU atende aos requisitos básicos, permitindo alterar o tempo de envio de mensagens para o *ThingsBoard*.

6. Conclusão e Trabalhos Futuros

Com base nos resultados obtidos durante o período de testes, constatou-se que o sistema operou de maneira estável e funcional ao longo de toda a sua utilização. Dessa forma, os objetivos definidos no início deste trabalho foram plenamente alcançados, demonstrando a viabilidade da solução proposta, mesmo diante dos desafios enfrentados durante o desenvolvimento.

Entre os principais desafios, destacaram-se a ausência de documentação adequada para alguns casos específicos, a dificuldade em encontrar bibliotecas estáveis para implementar o protocolo RNDIS e a busca por uma solução viável para permitir a reparametrização do *datalogger* CR-300. Apesar dessas dificuldades, as funcionalidades essenciais foram implementadas com sucesso, garantindo o atendimento aos requisitos do sistema.

Como trabalhos futuros, identificamos melhorias e novas funcionalidades que podem ser incorporadas ao sistema. Uma dessas melhorias é a implementação da reparametrização do *datalogger* CR-300, permitindo a alteração dinâmica das tabelas responsáveis pela coleta de dados. Além disso, é recomendada a adoção da comunicação via *Ethernet Over* USB, uma solução que simplificaria o processo de manutenção. Com essa abordagem, não seria necessário configurar registradores para cada novo sensor adicionado, bastando realizar uma requisição (*GET*) para obter os dados em formato *JSON* diretamente da tabela. Outra funcionalidade que pode ser implementada é a comunicação via LoRa, possibilitando que, no futuro, as estações não dependam exclusivamente de redes de comunicação como 4G ou Wi-Fi.

Essas evoluções ampliam a flexibilidade, a eficiência e a escalabilidade do sistema, tornando-o ainda mais robusto e prático para futuras aplicações.

Referências

Abdul Hafeez, P., Singh, G., Singh, J., Prabha, C., e Verma, A. (2022). IoT in Agriculture and Healthcare: Applications and Challenges. *3rd International Conference on Smart Electronics and Communication (ICOSEC)*, pages 446–450.

ANATEL (2024). Disponível em: https://www.gov.br/anatel/pt-br/assuntos/noticias/proposta-para-restringir-a-certificacao-de-equipamentos-que-utilizem-apenas-2g-ou-3g-esta-em-analise-1. Acessado em: 07 de dezembro 2024.

Ayaz, M., Ammad-Uddin, M., Sharif, Z., Mansour, A., e Aggoune, E.-H. M. (2019). Internet-of-Things (IoT)-Based Smart Agriculture: Toward Making the Fields Talk. *IEEE Access*, 7:551–583.

Bouaouad, A.-E., Cherradi, A., Assoul, S., e Souissi, N. (2020). The key layers of IoT architecture. In *5th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech)*, pages 1–4.

Faria, J. P. e et al. (2022). Development of an IoT communication module for energy sharing communities management and monitorization. In *IEEE International Conference on Environment and Electrical Engineering and 2022 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe)*, pages 1–5.

Fetahu, L., Maraj, A., e Havolli, A. (2022). Internet of Things (IoT) benefits, future perspective, and implementation challenges. In *45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)*, pages 399–404.

Junior, S. e Luiz, S. (2018). *Internet das coisas : fundamentos e aplicações em Arduino e NodeMCU*. Érica, São Paulo.

Kadir, E. A., Efendi, A., e Rosa, S. L. (2018). Application of LoRa WAN Sensor and IoT for Environmental Monitoring in Riau Province Indonesia. In *5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, pages 281–285.

Karthiga, R., Devi, C. L. B., Janaki, R., Gayathri, C., Pandi, V. S., e Shobana, D. (2023). IoT Farm: A Robust Methodology Design to Support Smart Agricultural System Using Internet of Things with Intelligent Sensors Association. In *7th International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, pages 1332–1337.

Khan, S. U., Zaheer, R., e Khan, S. (2012). Future Internet: The Internet of Things architecture, possible applications and key challenges. *International Conference on Frontiers of Information Technology (FIT)*, 10:257–260.

Kuo, L.-S., Chen, I.-H., e Peng, S.-H. (2022). Development and In-situ Teaching of Disaster-resistant Monitoring System on Slopeland with Micro-weather Stations and Microcontrollers. In *IEEE 5th Eurasian Conference on Educational Innovation (ECEI)*, pages 51–54.

Modbus Organization (2024). About Modbus. Disponível em: `https://www.modbus.org/`. Acessado em: 01 novembro 2024.

MQTT Organizatation (2022). Why MQTT? Disponível em: `https://mqtt.org`. Acesso em 01 maio 2024.

OASIS (2019). MQTT Version 5.0. Disponível em: `https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html`. Acessado em: 01 maio 2024.

Odida, M. (2024). The Evolution of Mobile Communication: A Comprehensive Survey on 5G Technology. *Journal of Sensor Networks and Data Communications*.

Oliveira, S. d. (2017). *Internet das Coisas com ESP8266, Arduino e Raspberry Pi*. Novatec, São Paulo.

Pinaculo (2019). O Leilão da 5G e diferenças entre 2G, 3G, 4G e 5G. Disponível em: https://pinaculo.com.br/blog/2021/11/18/o-leilao-da-5g-e-diferencas-entre-2g-3g-4g-e-5g/. Acessado em: 12 maio 2024.

Pyingkodi, M., Thenmozhi, K., Nanthini, K., Karthikeyan, M., Palarimath, S., Erajavignesh, V., e Kumar, G. A. (2022). Sensor Based Smart Agriculture with IoT Technologies: A Review. In *International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–7.

Rajendran, S., Jadhav, S. A., Praba, J. A., Muthukumaran, D., Kiran., K., e Sharma, S. (2023). Leveraging the Internet of Things (IoT) for Disaster Management: Enhancing Resilience, Early Warning System in a Globally Connected World. In *9th International Conference on Smart Structures and Systems (ICSSS)*, pages 1–6.

Silva, E. L. e Menezes, E. M. (2005). *Metodologia da Pesquisa e Elaboração de Dissertação*. Universidade Federal de Santa Catarina.

Singh, S. (2017). 1g, 2g,... & 5g: The evolution of the g's. *https://mse238blog.stanford.edu*.

Sun, L. e Wu, J. (2020). Application of 4G IOT Card in Automatic Weather Station. *Journal of Physics: Conference Series*, 1601(3):032049.

The Things Network (2022). What are LoRa and LoRaWAN? Disponível em: https://www.thethingsnetwork.org/docs. Acessado em: 03 maio 2024.

Thingsboard (2024). What is ThingsBoard? Disponível em https://thingsboard.io/docs/getting-started-guides/what-is-thingsboard/ Acessado em: 21 de novembro 2024.

Łostowski, A., Wilczek, A., Kafarski, M., Lewafndowski, A., Szypłowska, A., Skierucha, W., e Abramowicz, M. (2020). Wireless IoT communication module with low power consumption for a soil moisture and salinity sensor. In *Baltic URSI Symposium (URSI)*, pages 33–37.