UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Curso de Bacharelado em Ciência da Computação

Trabalho de Conclusão de Curso

Investigação de sensores virtuais tolerantes à falha para o monitoramento do uso de salas no contexto de um campus inteligente

Emerson de Vasconcelos Vieira

Pelotas, 2023

Emerson de Vasconcelos Vieira

Investigação de sensores virtuais tolerantes à falha para o monitoramento do uso de salas no contexto de um campus inteligente

Trabalho de Conclusão de Curso apresentado ao Centro de Desenvolvimento Tecnológico da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Rafael Iankowski Soares

Pelotas, 2023

Universidade Federal de Pelotas / Sistema de Bibliotecas
Catalogação na Publicação

V657i Vieira, Emerson de Vasconcelos

Investigação de sensores virtuais tolerantes à falha para o monitoramento do uso de salas no contexto de um campus inteligente / Emerson de Vasconcelos Vieira ; Rafael Iankowski Soares, orientador. — Pelotas, 2023.

70 f.

Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) — Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 2023.

1. Internet das coisas. 2. Sensor virtual. 3. Campus inteligente. I. Soares, Rafael Iankowski, orient. II. Título.

CDD : 005

Elaborada por Simone Godinho Maisonave CRB: 10/1733

Emerson de Vasconcelos Vieira

Investigação de sensores virtuais tolerantes à falha para o monitoramento do uso de salas no contexto de um campus inteligente

Trabalho de Conclusão de Curso aprovado, como requisito parcial, para obtenção do grau de Bacharel em Ciência da Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas.

Data da Defesa: 22 de Setembro de 2023

Banca Examinadora:

Prof. Dr. Rafael Iankowski Soares (orientador)
Doutor em Computação pela Pontifícia Universidade Católica do Rio Grande do Sul

Prof. Dr. Julio Carlos Balzano De Mattos
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Me. João Inácio Moreira Bezerra
Mestre em Computação pela Universidade Federal de Pelotas

Dedico ao meu querido avô Martiniano Vieira, que nos deixou dias antes da conclusão deste documento. Seu espírito inspirador viverá em minha memória e coração para sempre.

AGRADECIMENTOS

Agradeço, em primeiro lugar, a mim mesmo, bem como a toda a minha família, aos meus pais, Rosenilda e Gilberto, e aos meus irmãos, Giovani e Kelvin, por todo o apoio, incentivo, motivação e por estarem sempre ao meu lado.

Expresso minha sincera gratidão aos meus amigos, tanto aqueles que foram meus colegas durante a graduação quanto os que fiz ao participar do Grupo PET Computação, e também aos amigos que conquistei fora da faculdade.

Por fim, agradeço aos meus professores, em especial ao meu orientador, o Professor Rafael.

RESUMO

VIEIRA, Emerson de Vasconcelos. **Investigação de sensores virtuais tolerantes à falha para o monitoramento do uso de salas no contexto de um campus inteligente**. Orientador: Rafael Iankowski Soares. 2023. 70 f. Trabalho de Conclusão de Curso (Ciência da Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2023.

A evolução tecnológica, impulsionada pela capacidade de processamento e comunicação entre dispositivos computacionais, tem dado origem à Internet das Coisas (IoT), permitindo a automatização de serviços e o surgimento da Indústria 4.0. No contexto de IoT surge o conceito de sensor virtual, que pode simular a funcionalidade de um ou mais sensores físicos monitorando uma mesma variável do ambiente ou diferentes variáveis correlacionadas de modo que seja possível inferir informações em caso de falha de um sensor físico a fim de tornar o sistema tolerante a falhas. Atualmente na UFPel, mais especificamente no Campus Porto (Anglo), identifica-se a necessidade de melhorar o serviço de supervisão e gerenciamento das salas de aula e demais espaços acadêmicos ao longo do semestre. O controle e gestão atual das salas não permite a identificação de salas não utilizadas ou pouco utilizadas ao durante o semestre letivo causando uma má utilização dos recursos disponíveis. Esses problemas afetam a qualidade do ensino, pois as alocações acabam concentradas em alguns horários inadequadamente, resultando em conflitos de agendamento e dificuldades de acesso a recursos para atividades eventuais, mesmo quando os recursos estão subutilizados. Portanto, esse trabalho tem como objetivo desenvolver um sistema IoT tolerante a falhas explorando o conceito de sensor virtual para monitorar a ocupação de salas de aula e laboratórios no Campus Anglo. O uso de sensores virtuais permite que a partir de um conjunto de sensores físicos monitorando diferentes variáveis do ambiente seja possível identificar em tempo real a ocupação ou não dos espaços acadêmicos. O fato de haver diferentes variáveis do ambiente sendo monitoradas simultaneamente permite que o sensor virtual, a partir da definição de um algoritmo proposto, infira sobre a ocupação da sala, mesmo com a falha de algum sensor físico. O trabalho inicialmente investiga algumas variáveis do ambiente que podem indicar a ocupação da sala para atividades acadêmicas a partir dos sensores de luminosidade, de ruido sonoro, movimento de pessoas e até mesmo o ultrassônico para avaliação de distância. Em seguida, é proposto um algoritmo capaz de a partir dos resultados individuais dos sensores decidir sobre a ocupação ou não do ambiente. Os resultados obtidos pelo monitoramento de um laboratório de pesquisa mostram a eficiência da solução proposta mesmo com a ausência de uma variável do ambiente.

Palavras-chave: Internet das coisas. Sensor virtual. Campus inteligente.

ABSTRACT

VIEIRA, Emerson de Vasconcelos. **Investigation of fault-tolerant virtual sensors for monitoring room use of rooms in the context of a smart campus**. Advisor: Rafael Iankowski Soares. 2023. 70 f. Undergraduate Thesis (Computer Science) – Technology Development Center, Federal University of Pelotas, Pelotas, 2023.

Technological evolution, driven by the processing and communication capability between computing devices, has given rise to the Internet of Things (IoT), enabling the automation of services and the emergence of Industry 4.0. In the context of IoT, the concept of a virtual sensor arises, which can simulate the functionality of one or more physical sensors by monitoring the same environmental variable or different correlated variables, making it possible to infer information in case of a physical sensor failure in order to make the system fault-tolerant. Currently, at UFPel, specifically in the Campus Porto (Anglo), there is a need to improve the supervision and management of classrooms and other academic spaces throughout the semester. The current control and management of classrooms does not allow for identifying unused or underused classrooms during the academic semester, causing poor use of available resources. These problems affect the quality of teaching, as allocations are inappropriately concentrated in some timetables, resulting in scheduling conflicts and difficulties in accessing resources for occasional activities, even when resources are underutilized. Therefore, this work aims to develop a fault-tolerant IoT system exploiting the virtual sensor concept to monitor the occupancy of classrooms and laboratories at the Anglo Campus. The use of virtual sensors allows for the real-time identification of occupancy or vacancy in academic spaces based on a set of physical sensors monitoring various environmental variables. The fact that different environmental variables are being monitored simultaneously means that the virtual sensor, based on the definition of a proposed algorithm, can tell if the room is occupied, even if a physical sensor fails. The work initially investigates some environmental variables that can indicate room occupancy for academic activities using light sensors, sound noise sensors, people's movements, and even ultrasonic sensors for distance assessment. Next, an algorithm is proposed, which, based on the individual results of the sensors, can decide whether or not to occupy the room. The results obtained by monitoring a research laboratory show the proposed solution's efficiency even without an environmental variable.

Keywords: Internet of things. Virtual sensor. Smart campus.

LISTA DE FIGURAS

Figura 1 Estimativa do crescimento anual de IoT. Disponível em (AYDOS; VURAL; TEKEREK, 2019). 19
Figura 2 Exemplo de sensores virtuais. Disponível em Martin; Kühl; Satzger (2021) 22
Figura 3 Exemplo arquitetura MQTT. Disponível em (SONI; MAKWANA, 2017) 23

Figura 4 Diagrama do sistema proposto. 27
Figura 5 Módulo ESP32-wroom-32d. Disponível em: ESP32 SERIES - DATASHEET (2023) 28
Figura 6 Sensor de luz. Fonte: GROVE - LIGHT SENSOR (2016) 28
Figura 7 Sensor de som. Fonte: GROVE - SOUND SENSOR (2016) 29
Figura 8 Módulo Sensor Hc-sr501 de Presença e Movimento – PIR. Fonte: MóDULO SENSOR HC-SR501 DE PRESENçA E MOVIMENTO – PIR (2020) 29
Figura 9 Sensor Ultrassônico - HC-SR04. Fonte: O QUE é UM SENSOR ULTRASSôNICO (2023) 30
Figura 10 Exemplo de mensagem formatada como JSON com dados dos sensores. 32
Figura 11 Código em Python do subscriber 33
Figura 12 Dados do dia 04/08/2023 - Período específico. Fonte: Própria . . . 34

Figura 13 Esboço do leiaute da sala 337 e posicionamento do sistema de monitoramento. Fonte: Própria 36
Figura 14 Dados do dia 10/07/2023-1. Fonte: Própria 37
Figura 15 Dados do dia 10/07/2023-2. Fonte: Própria 38
Figura 16 Dados do dia 10/07/2023-3. Fonte: Própria 39
Figura 17 Dados do dia 10/07/2023 - Período específico. Fonte: Própria . . . 40
Figura 18 Dados do dia 03/08/2023-1. Fonte: Própria 41
Figura 19 Dados do dia 03/08/2023-2. Fonte: Própria 42
Figura 20 Dados do dia 21/08/2023-1. Fonte: Própria 43
Figura 21 Fluxograma do Sensor Virtual 47
Figura 22 Código em C++ das variáveis e função para o cálculo da média dinâmica do Sensor Ultrassônico 47
Figura 23 Código em C++ do Sensor Virtual 48
Figura 24 Dados do dia 22/08/2023-1. Fonte: Própria 50
Figura 25 Dados de teste do Sensor Virtual com 30s do dia 22/08/2023-1. Fonte: Própria 50
Figura 26 Dados do dia 31/08/2023-1. Fonte: Própria 51

Figura 27 Dados de teste do Sensor Virtual com 120s do dia 31/08/2023-1. Fonte: Própria 52
Figura 28 Dados do dia 01/09/2023-1. Fonte: Própria 53
Figura 29 Dados de teste do Sensor Virtual com 900s do dia 01/09/2023-1. Fonte: Própria 53
Figura 30 Dados do dia 01/09/2023-2. Fonte: Própria 54
Figura 31 Dados de teste do Sensor Virtual com 900s do dia 01/09/2023-2. Fonte: Própria 54
Figura 32 Dados do dia 01/09/2023-3. Fonte: Própria 55
Figura 33 Dados de teste do Sensor Virtual com 900s do dia 01/09/2023-3. Fonte: Própria 55
Figura 34 Dados do dia 01/09/2023-4. Fonte: Própria 57
Figura 35 Dados do Sensor Virtual do dia 01/09/2023-4. Fonte: Própria 57
Figura 36 Dados do dia 05/09/2023-1. Fonte: Própria 59
Figura 37 Dados de teste de tolerância a falhas do Sensor Virtual do dia 05/09/2023-1. Fonte: Própria 60
Figura 38 Dados do dia 06/09/2023-2. Fonte: Própria 61
Figura 39 Dados de teste de tolerância a falhas do Sensor Virtual do dia 06/09/2023-2. Fonte: Própria 62
Figura 40 Dados do dia 06/09/2023-1. Fonte: Própria 63
Figura 41 Dados de teste de tolerância a falhas do Sensor Virtual do dia 06/09/2023-1. Fonte: Própria 64

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CPU	Central Processing Unit
IoT	Internet of Things
ICT	Information and Communication Technology
MQTT	Message Queuing Telemetry Transport
PIR	Passive Infrared
LDR	Light Dependent Resistor
JSON	JavaScript Object Notation

SUMÁRIO

1 INTRODUÇÃO 15

2 REFERENCIAL TEÓRICO 18
2.1 Internet das coisas 18
2.2 Computação em borda 19
2.3 Indústria 4.0 20
2.4 Ambientes inteligentes 20
2.5 Sensor Físico 21
2.6 Sensor virtual 21
2.7 Protocolo MQTT 22
2.8 Revisão Bibliográfica 24

3 SISTEMA DE MONITORAMENTO PROPOSTO 26
3.1 Hardware utilzado 27
3.1.1 ESP32 27
3.1.2 Sensor de luminosidade 28
3.1.3 Sensor sonoro 28
3.1.4 Módulo Sensor de Presença e Movimento – PIR 29
3.1.5 Sensor Ultrassônico - HC-SR04 30
3.2 Definição do Broker MQTT 30
3.3 Aquisição dos dados 30
3.4 Configuração do Cliente Subscriber para recebimento e armazenamento dos dados 32

4 RESULTADOS OBTIDOS 35
4.1 Análise dos sensores: dados produzidos 35
4.2 Desenvolvimento do Sensor Virtual 44
4.2.1 Cenário 1: com luz ligada 45
4.2.2 Cenário 2: com luz desligada 46
4.3 Cenário 1: com luz ligada 49
4.4 Cenário 2: com luz desligada 56
4.5 Cenários de falhas nos sensores 58
4.5.1 Falha no sensor ultrassônico 58
4.5.2 Falha no sensor PIR 60
4.5.3 Falha no sensor LDR 62

5 CONCLUSÕES E TRABALHOS FUTUROS 65

REFERÊNCIAS 67

1 INTRODUÇÃO

A evolução tecnológica tem permitido uma maior capacidade de processamento de informações e comunicação entre dispositivos computacionais. Além disso, a produção em larga escala de circuitos integrados viabilizou também que objetos possam se comunicar por meio de uma rede sem fio produzindo um grande volume de dados sobre um dado ambiente sem a intervenção direta do ser humano. Esta rede de comunicação de objetos é conhecida como Internet das Coisas (em inglês, Internet of Things - IoT) e permite a automatização de serviços por meio de sensores e atuadores interagindo por meio de algoritmos, dando origem a Indústria 4.0 (KUMAR et al., 2023), casas, prédios e até mesmo cidades inteligentes (BENEDICT; RUMAISE; KAUR, 2019).

A Indútria 4.0 que de acordo com Santos et al. (2018), "Abrange um conjunto de tecnologias de ponta ligadas à Internet com objetivo de tornar os sistemas de produção mais flexíveis e colaborativos.", tem impulsionado o desenvolvimento de novas soluções tecnológicas em diversos setores. Dentre essas soluções, estão, como citado anteriormente os ambientes inteligentes, como as casas inteligentes, os prédios inteligentes e as cidades inteligentes.

Esses ambientes inteligentes, também conhecidos como espaços inteligentes, são caracterizados por integrar diversos dispositivos conectados a rede, com a capacidade de processamento. Segundo Farias et al. (2019), "Um espaço inteligente pode ser caracterizado como um ambiente com diversos dispositivos, conectados em rede, os quais possuem capacidade de processamento e sensoriamento e que auxiliam os usuários finais na execução de suas tarefas de forma mais eficiente."

Neste contexto surge o conceito de sensor virtual, que pode simular a funcionalidade de um ou mais sensores físicos monitorando uma mesma variável do ambiente ou diferentes variáveis correlacionadas de modo que seja possível inferir informações em caso de falha de um sensor físico a fim de tornar o sistema tolerante a falhas. No entanto, os sensores virtuais não substituem completamente os sensores físicos. Eles são utilizados para minimizar a tarefa dos sensores físicos, delegando a maioria das tarefas ao software, representado como sensores virtuais, o que proporciona cone-

xões remotas entre regiões de interesse e a correção de erros, reduzindo a tarefa dos sensores físicos (GUPTA; MUKHERJEE, 2016).

Um campus inteligente é uma tendência emergente que permite às instituições de ensino combinar tecnologias inteligentes para obter uma melhor eficiência no uso de recursos como água, energia elétrica, controle de segurança, conforto térmico de ambientes entre outros serviços. Os ambientes inteligentes possuem a infraestrutura física necessária para fornecer serviços, tomada de decisões, gerenciamento de recursos, controles de segurança e muito mais (MIN-ALLAH; ALRASHED, 2020).

Na realidade do Campus Anglo da Universidade Federal de Pelotas (UFPel), foi identificada a necessidade de melhorar a supervisão e utilização das salas de aula e demais espaços acadêmicos. Atualmente, o uso da sala é controlado manualmente, o que pode levar a dificuldades na identificação de salas não utilizadas, conflitos de agendamento, o que caracteriza um problema no gerenciamento no gerenciamento destes ambientes e consequentemente o uso inadequado dos recursos disponíveis.

A alocação indevida de horários nos espaços acadêmicos pode causar conflito de escala, dificuldade de agendamento das aulas e atividades, bem como dificuldade de acesso aos recursos necessários para a realização de atividades acadêmicas, assim afetando diretamente a qualidade de ensino. O que acontece muitas vezes é professores agendam em um mesmo horário uma sala de aula e um laboratório visando a necessidade de disciplinas com carga horária teórica e prática e só utilizarem um desses espaços. Outro problema recorrente é a sobrecarga de algumas salas de aulas. Assim se tornando necessário ter informações sobre quais ambientes acadêmicos estão sendo utilizados em determinados horários durante o semestre para se ter o controle da ocupação desses ambientes.

Estes ambientes são escassos e mal gerenciados na UFPel visto que o controle da alocação destes espaços não é realizado de modo eficiente, o que resulta frequentemente na indisponibilidade de salas para atividades extracurriculares, mesmo havendo salas desocupadas. Isso ocorre porque muitas salas são alocadas para aulas desde o início do semestre, embora não sejam utilizadas ao longo de todo o período, principalmente devido as disciplinas teórico-práticas.

Além dos problemas mencionados anteriormente, a alocação inadequada de espaços acadêmicos pode gerar muitos outros problemas, como por exemplo uma utilização ineficiente de recursos, como a iluminação, ventilação, refrigeração e equipamentos de laboratório, resultando em um desperdício de energia da universidade. Além disso, a coleta de dados sobre a ocupação dos ambientes acadêmicos pode ser usada para aprimorar a gestão dos espaços e planejar futuras alocações de acordo com a demanda.

Tendo em vista esses problemas, esse trabalho tem como objetivo desenvolver um sistema IoT tolerante a falhas explorando o conceito de sensor virtual para monitorar

a ocupação de salas de aula e laboratórios no Campus Anglo. Para isso, são investigadas diferentes variáveis físicas do ambiente por meio de sensores físicos tais como luz, ruído e presença de pessoas, por exemplo, para o monitoramento da ocupação destes ambientes, com o objetivo de inferir as informações desejadas mesmo na falha de algum dos sensores físicos. Outro aspecto importante na implementação do trabalho é avaliar as posições dos sensores em diferentes pontos da sala e analisar a relação das informações geradas e o contexto analisado. Além disso, será adotada uma abordagem de computação em borda, permitindo que o processamento e análise dos dados aconteçam diretamente no dispositivo de borda.

Esta monografia está organizada em 5 capítulos, descritos a seguir. No Capítulo 2 está um referencial teórico detalhado, onde são apresentados os principais conceitos utilizados neste estudo, junto com a revisão bibliográfica, que apresenta alguns trabalhos relacionados. No Capítulo 3, está apresentado o desenvolvimento do sistema de monitoramento proposto, abordando as decisões tomadas para a sua execução. O Capítulo 4 traz os resultados obtidos ao longo deste estudo, incluindo a análise e interpretação dos dados. Por fim, o Capítulo 5 conclui o trabalho, apresentando as conclusões obtidas, e as possíveis aplicações futuras que podem ser exploradas em próximos estudos.

2 REFERENCIAL TEÓRICO

Neste capítulo são revisados os principais conceitos necessários para o entendimento da solução proposta. Inicialmente é apresentada as principais definições de Internet da Coisas. Em seguida, é apresentado o conceito de Computação em Borda, usada principalmente por centralizar o processamento de dados em aplicações IoT. Além disso, revisam-se outros conceitos como a importante área que impulsionou o uso de IoT, a Indústria 4.0 e que hoje segue como uma nova revolução deste setor produtivo com base em tecnologias como a inteligência artificial, por exemplo. Além disso, são revisados os conceitos de sensores e sensores virtuais e suas vantagens no contexto de IoT.

2.1 Internet das coisas

A definição de IoT pode ser um pouco abstrata, pois ela tem várias definições que variam dependendo do contexto e da perspectiva em que é abordada, mas o que seria a melhor definição de Internet das Coisas tentando levar em consideração todas abordagens. Segundo Madakam et al. (2015), IoT é “Uma rede aberta e abrangente de objetos inteligentes que possuem a capacidade de se auto-organizar, compartilhar informações, dados e recursos, reagindo e agindo a situações e mudanças no ambiente”.

IoT descreve um mundo em que praticamente tudo pode estar conectado à Internet e se comunicar de maneira inteligente como nunca visto antes. Embora muitos de nós relacionemos o termo "estar conectado"apenas a dispositivos eletrônicos como servidores, computadores, tablets, telefones e smartphones, a verdade é que sensores e atuadores embutidos em objetos físicos também podem estar conectados por meio de redes com e sem fio (MADAKAM et al., 2015).

“A Internet das Coisas é uma revolução tecnológica que representa o futuro da computação e das comunicações” (MADAKAM et al., 2015), essa revolução tecnológica que a IoT proporcionou serviu de impulso para a indústria e impactando em uma nova revolução industrial, a quarta revolução industrial ou Indústria 4.0.

IoT não é apenas uma única tecnologia, é a combinação de diferentes tecnologias de hardware e software, ela fornece soluções com base na integração da tecnologia da informação abrangendo tanto o hardware e software utilizados para armazenar, recuperar e processar dados e à tecnologia de comunicação que inclui sistemas eletrônicos usados para a comunicação entre indivíduos ou grupos (PATEL; PATEL; SCHOLAR, 2016). Aplicações IoT estão crescendo diariamente expandindo áreas de aplicações de modo a trazer conforto para as pessoas, reduzir custos e melhorar a produtividade. Aydos; Vural; Tekerek (2019) apresentam na Figura 1 uma estimativa de crescimento de IoT, o que motiva ainda mais as pesquisas e desenvolvimento de sistemas apoiados em objetos inteligentes.

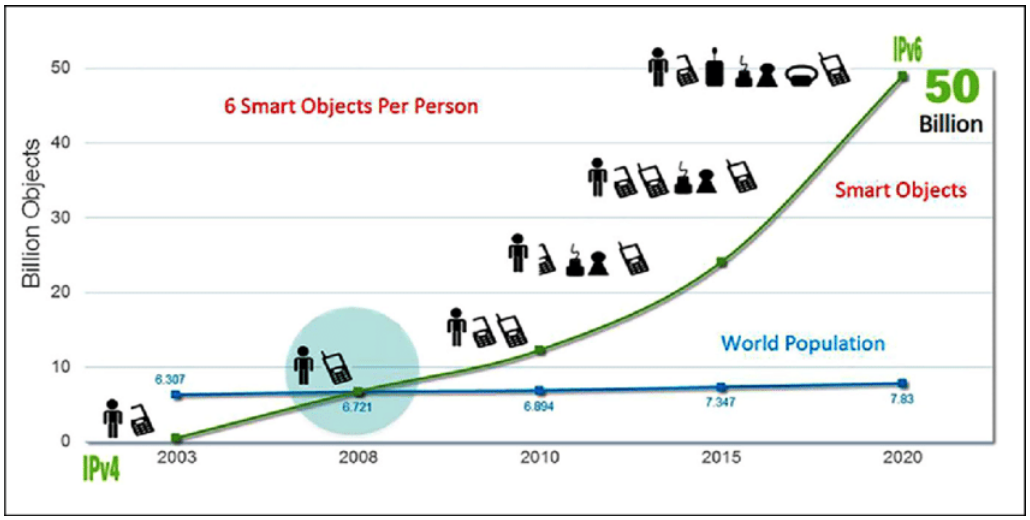

Figura 1 – Estimativa do crescimento anual de IoT. Disponível em (AYDOS; VURAL; TEKEREK, 2019).

2.2 Computação em borda

Gupta; Mukherjee (2019) discutem os desafios de integrar sensores virtuais com computação em nuvem, mais precisamente na integração da rede de dispositivos IoT usando o conceito de sensores virtuais e computação em borda (do inglês, *Edge Computing*) (SHI et al., 2016), a computação em borda permite o processamento de dados da rede de dispositivos IoT que adquirem dados por meio de sensores e que estão próximos ao servidor de borda, a fim de reduzir a quantidade de transferência dados entre dispositivos IoT e a nuvem, que normalmente possui alto custo em termos de latência.

Segundo Sittón-candanedo et al. (2019), as arquiteturas de computação em borda oferecem uma solução às infraestruturas IoT porque são capazes de gerenciar os dados heterogéneos gerados por dispositivos IoT. Com isso as arquiteturas melhoram o desempenho, diminuem a latência, reduzem o custo de recursos e aumentam a capacidade de resposta, escalabilidade, confiabilidade, segurança e privacidade.

2.3 Indústria 4.0

Schwab (2019) destaca que em nossa história, revoluções têm surgido quando novas tecnologias e novas formas de perceber o mundo provocam uma transformação significativa nas configurações sociais e nos sistemas econômicos. Tendo isso em vista, a quarta e última revolução industrial, também denominada indústria 4.0, está sendo a mais tecnológica e digital de todas, impulsionada por fatores como IoT, inteligência artificial, computação em nuvem (em inglês, *cloud computing*) e entre outras muitas tecnologias, todas as quais estão conectadas à rede.

De acordo com Santos et al. (2018), a Indústria 4.0 representa uma evolução natural dos sistemas industriais precedentes, desde a mecanização do trabalho ocorrida no século XVIII até a automação da produção que vigora nos tempos contemporâneos. Nesse contexto de evolução tecnológica, a adoção do conceito de IoT em segmentos como cidades inteligentes, agricultura, segurança e saúde cresce rapidamente, aumentando a demanda por soluções que possam lidar com falhas, já que uma única falha pode levar a consequências imprevisíveis em ambientes IoT (PASTÓRIO; RODRIGUES; CAMARGO, 2020).

Pastório; Rodrigues; Camargo (2020) ainda afirmam que, para garantir a confiabilidade de um sistema, frequentemente recorrem-se a técnicas de tolerância a falhas, as quais têm como objetivo assegurar o correto funcionamento do sistema, mesmo na presença de falhas, sendo baseadas em redundância, o que implica a inclusão de componentes adicionais ou a aplicação de algoritmos específicos.

2.4 Ambientes inteligentes

Proporcionadas pela Indústria 4.0, as cidades inteligentes, assim como casas e prédios inteligentes, vêm recebendo um crescente investimento em busca de novas tecnologias para facilitar e melhorar a qualidade de vida das pessoas. De acordo com Yin et al. (2015), a definição geral de cidade inteligente é o uso de Tecnologia da Informação e Comunicação (em inglês, *Information and Communication Technology* - ICT) para tornar a cidade (administração, educação, transporte, etc.) mais inteligente e eficiente.

As cidades inteligentes surgem principalmente da evolução tecnológica aplicada inicialmente na indústria e em um segundo momento na automação de casas e prédios. A motivação principal é a busca por melhores soluções para redução de consumo energético, manutenção mais eficiente, segurança inclusive por acesso remoto em tempo real e principalmente melhor conforto e qualidade de vida para as pessoas.

2.5 Sensor Físico

Lotufo; Garcia (2008) descreve que qualquer sistema de controle ou monitoramento necessita da utilização de elementos de interface com o mundo real, ou mundo físico. Assim, um processo industrial requer uma diversidade de sensores para poder observar e identificar seu estado atual e poder tomar decisões de controle.

Segundo Gonzalez (1999), “sensores são os olhos pelos quais o comportamento de uma planta é visto e informação sobre seu desempenho é obtida”. Sensores físicos são dispositivos eletronicos que monitoram e medem variaveis de grandeza física, como temperatura, pressão, aceleração, força, entre outras. Eles transformam essa grandeza física em um sinal digital que pode ser interpretado por sistemas computacionais.

Os sensores têm o papel de transformar grandezas físicas em sinais digitais, porém isso pode resultar em alguns problemas, segundo Lotufo; Garcia (2008) esses problemas são:

1. Erros de Medição;
2. Disponibilidade e Confiabilidade;
3. Atrasos de Medição;
4. Distância do ponto de medição;
5. Ambiente de medição;
6. Interferência no processo;
7. Preço.

2.6 Sensor virtual

A introdução de novas tecnologias em nosso cotidiano traz consigo outros desafios e novos problemas como por exemplo, falhas na conexão da rede entre os objetos ou dispositivos sensores e atuadores, podendo assim comprometer a funcionalidade do sistema como um todo. Neste contexto, o uso de técnicas que permitem manter o sistema operando mesmo na ocorrência de falha permite a criação de sistemas tolerantes a falhas (MUDASSAR; ZHAI; LEJIAN, 2022).

Para este fim, surge a definição de sensores virtuais aplicados em IoT (MARTIN; KÜHL; SATZGER, 2021), ou seja, a possibilidade de replicação de sensores físicos, como por exemplo um sensor de temperatura que pode ser replicado ou mesmo sensores de variáveis similares como umidade e pressão atmosférica e que a partir de

modelos matemáticos e estatísticos pode-se inferir situações no ambiente (CRISTALDI et al., 2020).

No trabalho proposto por Cristaldi, o sensor virtual é obtido a partir de quatro diferentes tipos de sensores para inferir uma informação relevante para um sistema de controle em malha fechada aplicado. A Figura 2 apresenta um esboço das formas de conceber um sensor virtual, segundo Martin; Kühl; Satzger (2021).

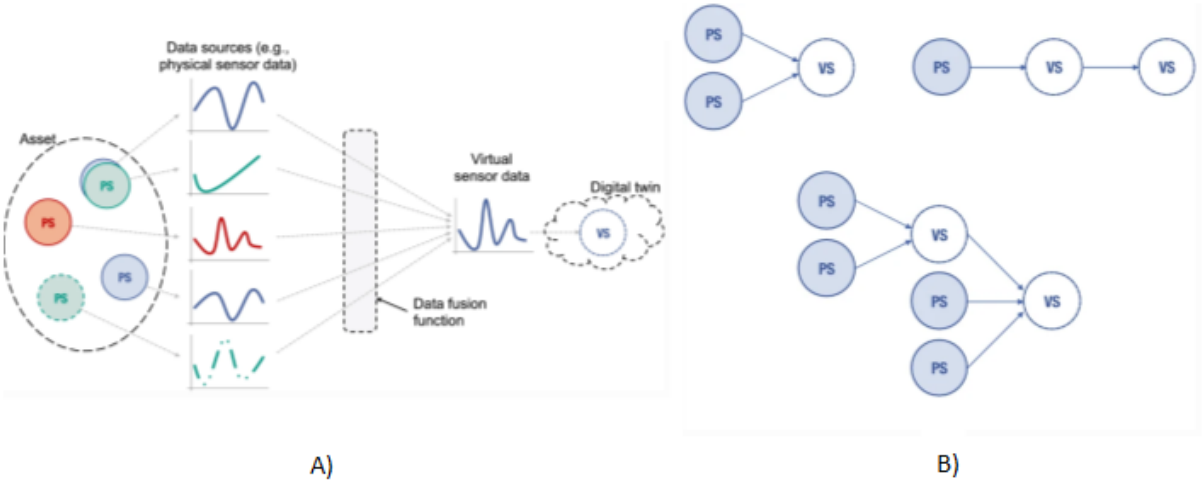

Figura 2 – Exemplo de sensores virtuais. Disponível em Martin; Kühl; Satzger (2021)

Assim, a falha de um destes sensores pode ser detectada e o sensor virtual permite manter o funcionamento do sistema por meio dos demais sensores em atividade. Os sensores virtuais podem ser aplicados em diferentes áreas da aplicação como por exemplo, cuidados com a saúde Raveendranathan et al. (2012), interação social e entretenimento Ping; Liu; Weng (2019).

2.7 Protocolo MQTT

O protocolo de comunicação *Push* é uma escolha adequada para dispositivos IoT, especialmente em situações onde a largura de banda da rede é limitada. Dentro do campo dos protocolos de comunicação para IoT, existem várias opções notáveis, como o MQTT, XMPP e CoAP, que foram desenvolvidos para suportar comunicações *push*. No entanto, o MQTT se destaca como uma solução amplamente adotada em uma variedade de dispositivos IoT e sistemas de mensagens instantâneas. Isso se deve ao fato de que o MQTT foi projetado especificamente para operar em dispositivos de baixo consumo de energia, sendo um protocolo leve e eficiente nesse contexto(SONI; MAKWANA, 2017).

O Protocolo MQTT (*Message Queuing Telemetry Transport*) é um protocolo de comunicação leve e eficiente que tem sido amplamente utilizado no campo de IoT para gerenciar a troca de dados entre os dispositivos devido também ao seu modo de assinatura e liberação. (CHEN et al., 2020).

O Protocolo MQTT foi introduzido pela IBM em 1999 e é um protocolo de comunicação que opera no modelo publish/subscribe padronizado, ou seja, segue os alguns conceitos básicos segundo Soni; Makwana (2017), onde os principais são esses:

1. Publicar/assinar: No protocolo MQTT, o publicador publica mensagens em tópicos específicos, enquanto os assinantes se inscrevem em tópicos específicos relacionados a eles e passam a receber todas as mensagens que são publicadas nesses tópicos.

2. Tópicos e assinaturas: O publicador envia mensagens em tópicos que podem ser considerados como assunto da mensagem. Já os assinantes se inscreve em tópicos para receber mensagens específicas. Os tópicos contêm dois níveis, ou seja, pode haver sub-tópicos, para obter informações abrangendo uma variedade de assuntos relacionados.

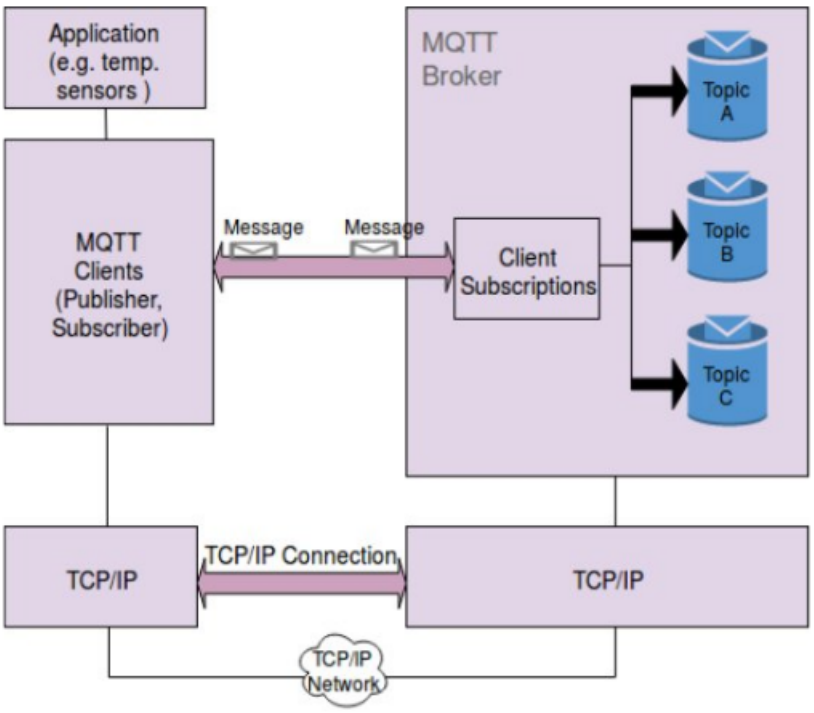

Figura 3 – Exemplo arquitetura MQTT. Disponível em (SONI; MAKWANA, 2017)

Além disso, o MQTT segue uma arquitetura típica que pode ser dividida em dois componentes principais, conforme ilustrado na Figura 3. A seguir, cada componente é brevemente descrito, conforme indicado por (SONI; MAKWANA, 2017):

1. Cliente: O cliente pode ser tanto um um publicador quanto um assinante, ou seja, ele pode publicar mensagens em tópicos e também se inscrever em tópicos de interesses para receber mensagens. O cliente é sempre quem estabelece a conexão de rede com o servidor(*Broker*).

2. Broker: O Broker, que funciona como o nosso servidor, controla a distribuição das mensagens e é responsável por receber todas as mensagens dos Publicadores em qualquer tópico, ele então filtra as mensagens e as envia para os Assinantes inscritos nos tópicos em questão.

2.8 Revisão Bibliográfica

Na revisão bibliográfica, apresentamos uma análise dos trabalhos publicados que se propõem a resolver problemas relacionados.

Cristaldi et al. (2020) propõem um exemplo de sensor virtual dedicado a monitorar diferentes seções de um inversor dedicado a aplicação fotovoltaica. Em particular, é mostrado como é possível monitorar a temperatura da CPU empregada na placa de controle, utilizando como base para definir o modelo dos sensores virtuais as temperaturas do fluxo de entrada e saída do gabinete, além da potência de saída. Essa solução ajuda a melhorar a confiabilidade do sistema aumentando a proporção de falhas detectadas com segurança, além disso, a utilização do sensor virtual reduz os custos de manutenção, eliminando a necessidade de redundância de hardware.

Outro trabalho relevante é proposto por Peniak; Bubeníková; Kanáliková (2021), que tem como foco principal a criação de um modelo de sensores virtuais redundantes, aplicando o conceito de gêmeos digitais que estão associados a sensores físicos, onde para a implementação a abordagem de computação de borda é selecionada, pois esta suporta requisitos com processamento em tempo real tempo e latência muito baixa. O dispositivo de borda deve ter um broker MQTT incorporado, que é usado para sua comunicação com sensores físicos MQTT, os sensores físicos são conectados ao broker MQTT do Edge e podem publicar seus resultados físicos medidos nos tópicos MQTT associados. O objetivo do trabalho é criar um modelo comum que permita uma abordagem genérica independente da implementação dos fornecedores.

Zhou et al. (2015) propõem o uso de tolerância a falhas em sistemas IoT por meio de uma arquitetura orientada a serviços utilizando sensores virtuais. O esquema proposto explora o fato dos sistemas IoT possuírem vários sensores físicos com diferentes propósitos, dependendo da aplicação, o que naturalmente proporciona redundância de informações desejadas para soluções tolerantes a falhas. Dessa forma, os autores apresentam serviços virtuais que utilizam dados de mais de um sensor físico, permitindo a substituição de um serviço real, ou seja, produzido por um conjunto pré-definido de elementos sensores físicos, caso ocorra falha em algum deles. O trabalho utiliza regressão linear para identificar e gerar os serviços virtuais a partir dos sensores físicos disponíveis. Dependendo do tipo de correlação existente entre os dados dos sensores utilizados para gerar o serviço, empregam-se mínimos quadrados recursivos (*recursive least squares* - RLS) ou regressão adaptativa multivariada (*multivariate*

adaptive regression splines - MARS) para a geração de sensores virtuais. A correlação entre os sensores é avaliada a partir do histórico de medições obtidas por meio de um protótipo

Em Gupta; Mukherjee (2017) são apresentados desafios de projeto de sistema IoT ao explorar características de sensores virtuais. Os autores investigam propriedades tais como clonagem, detecção de múltiplos eventos e abstrações as quais são efetivas em disponibilizar uma versão em software de sensores virtuais. Este trabalho discute um estudo de diferentes tipos de sensores virtuais, discute vantagens e experimentos com os sensores virtuais baseados em implementações preliminares.

Datta; Bonnet (2017) propõem um mecanismo para criar e operacionalizar um dispositivo virtual que pode representar um sensor ou atuador virtual. O trabalho apresenta uma motivação para o uso de sensores virtuais em IoT, entretanto a aplicação não está diretamente relacionada a aplicações tolerantes a falhas.

Casado-vara et al. (2018) propõem um algoritmo de controle cooperativo baseado em um modelo obtido a partir de cadeias de Markov para identificar e substituir sensores com falhas a fim de manter um sistema IoT tolerante a falhas. A proposta explora o conceito de sensores virtuais propondo um algoritmo para permitir manutenção e confiabilidade a sistemas desta natureza. Desta forma sensores com mal funcionamento são substituídos de acordo com o modelo de predição do algoritmo proposto a fim de corrigir o sistema.

Na revisão bibliográfica, foram analisados vários trabalhos relacionados que têm relevância direta para o projeto proposto em questão. Destacamos a importância dos sensores virtuais, que simulam as funcionalidades de sensores físicos, não apenas para aumentar a confiabilidade do sistema, mas também para reduzir os custos de manutenção. Além disso, identificamos a utilização crucial do protocolo MQTT e da computação em borda para possibilitar a comunicação e o processamento em tempo real. Além disso, encontramos estudos que se concentram na tolerância a falhas em sistemas IoT, aproveitando a redundância de informações de sensores físicos para manter a operação confiável. Essas pesquisas destacam a relevância dos sensores virtuais e seu potencial para aprimorar a eficiência e a confiabilidade dos sistemas IoT, o que está diretamente alinhado com os objetivos do nosso projeto.

3 SISTEMA DE MONITORAMENTO PROPOSTO

Este capítulo apresenta em detalhes a infraestrutura de hardware proposta, abordando as principais decisões de projeto tomadas para a definição do sistema de detecção do uso de salas baseado no uso de sensores virtuais. As decisões de projeto tomadas repercutem as questões de detecção de pessoas em uma sala no contexto de um campus universitário de modo a garantir também que o sistema seja tolerante a falhas de sensores físicos.

A Figura 4 apresenta o modelo do sistema IoT proposto, com os seguintes componentes:

1. **Broker MQTT:** Este é o servidor MQTT ao qual os clientes se conectarão. Os clientes podem se inscrever em tópicos para receber mensagens e publicar mensagens nos tópicos.

2. **ESP-32:** Este é um cliente do tipo *publisher*, que contém tanto sensores físicos quanto a aplicação do sensor virtual. Ele publica os dados desses sensores em tópicos específicos.

3. **Cliente Subscriber:** Este tipo de cliente é responsável por assinar os tópicos nos quais a ESP-32 publica as mensagens e armazenar essas informações.

Vale ressaltar que o diagrama apresentado contém apenas dois clientes, porém o sistema é capaz de suportar vários clientes conectados, permitindo a publicação de mensagens e a assinatura em vários tópicos.

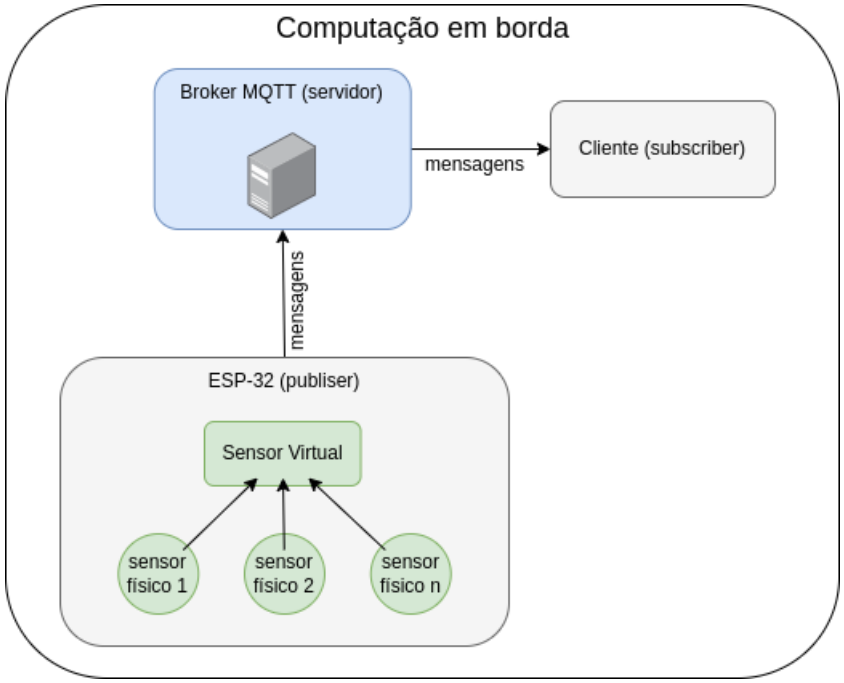

Figura 4 – Diagrama do sistema proposto.

3.1 Hardware utilzado

Neste trabalho, com o objetivo de coletar os dados necessários para o desenvolvimento do sensor virtual proposto e estabelecer a comunicação com um servidor de borda, foram escolhidos o microcontrolador para aquisição e transmissão de dados e os sensores físicos:

3.1.1 ESP32

A definição do microcontrolador está baseada no atendimento dos seguintes pré-requisitos necessários ao sistema de aquisição de dados: (i) ter conectividade sem fio via rede (Wi-Fi), rede comummente usada em toda a UFPel, permitindo comunicação eficaz com outros dispositivos, redes e a Internet; (ii) fácil programação via linguagem tipo C, tal como a usada na plataforma Arduino, a qual possui uma ferramenta de desenvolvimento livre e de fácil utilização; (iii) baixo custo financeiro e (iv) interface de comunicação digital e analógica para integração com grande diversidade de sensores. Neste sentido, o microcontrolador ESP32 (ESP32 SERIES - DATASHEET, 2023) atende a tais requisitos, por isso foi o escolhido como a principal infraestrutura de aquisição de dados dos sensores neste projeto.

Foi selecionado o microcontrolador ESP32, apresentado na Figura 5, devido à sua capacidade de atender aos requisitos deste projeto. Este dispositivo oferece conectividade sem fio (Wi-Fi), permitindo comunicação eficaz com outros dispositivos, redes e a internet. Além disso, ele é uma opção de custo acessível e é compatível com a plataforma Arduino, o que falicita a programação e a integração de sensores no

projeto.

Figura 5 – Módulo ESP32-wroom-32d. Disponível em: ESP32 SERIES - DATASHEET (2023)

3.1.2 Sensor de luminosidade

A Figura 6 apresenta o módulo do sensor de luz disponível no kit de desenvolvimento Grove da Seeed Studio (GROVE - LIGHT SENSOR, 2016), disponível na UFPel. Este módulo contém um resistor dependente de luz (em inglês, Light Dependent Resistor - LDR) que, de acordo com (WIKIPEDIA CONTRIBUTORS, 2023) "É um componente passivo que diminui a resistência em relação à recepção de luminosidade (luz) na superfície sensível do componente".

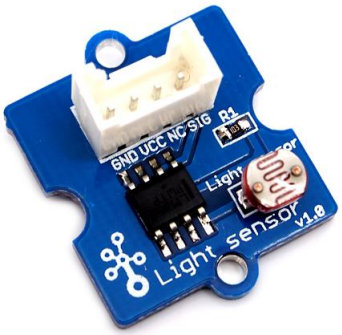

Figura 6 – Sensor de luz. Fonte: GROVE - LIGHT SENSOR (2016)

3.1.3 Sensor sonoro

A Figura 7 apresenta o sensor de som disponível no kit de desenvolvimento Grove (GROVE - SOUND SENSOR, 2016), que permite "detectar a intensidade sonora do ambiente". O principal componente do módulo é uma cápsula de eletreto, cujo sinal elétrico gerado é amplificado por um amplificador operacional LM358. Este módulo funciona como um microfone, transformando as ondas sonoras em sinais elétricos.

Figura 7 – Sensor de som. Fonte: GROVE - SOUND SENSOR (2016)

3.1.4 Módulo Sensor de Presença e Movimento – PIR

A Figura 8 apresenta o módulo Sensor Hc-sr501, que é capaz de detectar presença e movimento. Um sensor de movimento PIR consiste principalmente em um sensor piroelétrico que detecta a radiação infravermelha e uma lente óptica que concentra a radiação infravermelha no sensor. Os sensores PIR detectam mudanças na radiação infravermelha (calor radiante) medindo a temperatura dos objetos e as características da superfície em seu campo de visão. O termo ”passivo” em PIR refere-se ao fato de que esses dispositivos não emitem energia para detectar movimento. Este módulo permite detectar movimentos, sendo mais frequentemente usado para determinar quando um ser humano se move dentro de uma faixa de detecção (FUNCIONAMENTO DO SENSOR DE PRESENÇA PIR, 2018). O sensor PIR é coberto pela Lente *Fresnel*, que aumenta o alcance e compõe a parte externa do módulo sensor.

Figura 8 – Módulo Sensor Hc-sr501 de Presença e Movimento – PIR. Fonte: MóDULO SENSOR HC-SR501 DE PRESENçA E MOVIMENTO – PIR (2020)

3.1.5 Sensor Ultrassônico - HC-SR04

A Figura 9 apresenta o sensor Ultrassônico - HC-SR04, que é um dispositivo que utiliza a alta frequência de som para medir a distância entre itens determinados, sendo que também pode ser usado para detectar a presença. Este tipo de sensor consiste de uma unidade de transceptor único que emite pulsos de som inaudíveis, ele mede o tempo que esses pulsos levam para refletir em objetos, usando um temporizador para calcular distâncias precisas (O QUE É UM SENSOR ULTRASSÔNICO, 2023).

Figura 9 – Sensor Ultrassônico - HC-SR04. Fonte: O QUE é UM SENSOR ULTRASSôNICO (2023)

3.2 Definição do Broker MQTT

Com o avanço da Internet das Coisas, as formas de comunicação e os protocolos começaram a exigir novas formas de ações, que permitam conexões mais rápidas, leves e eficientes entre os objetos. O broker é como uma espécie de mediador entre as máquinas, capaz de fazer com que a comunicação de fato ocorra entre elas, assim tornando-se o elemento responsável por gerir as publicações e as subscrições por meio do protocolo MQTT.

Como broker MQTT, optou-se pelo *Eclipse Mosquitto*, um intermediário de mensagens de código aberto licenciado sob EPL/EDL, que implementa as versões 5.0, 3.1.1 e 3.1 do protocolo MQTT. O *Mosquitto* é leve e adequado para uso em uma ampla variedade de dispositivos, desde computadores de placa única de baixo consumo até servidores completos.(LIGHT, 2017)

Adicionalmente, o *Mosquitto* é especialmente apropriado para o nosso projeto, pois permite a criação de uma instância local, viabilizando a computação em borda e reduzindo a latência das mensagens publicadas. Além disso, é altamente portátil, com suporte para diversas plataformas, incluindo Windows, Mac, Linux e Raspberry Pi, o que simplifica e generaliza a implementação do projeto.

3.3 Aquisição dos dados

Os estudos de caso para prova de conceito deste projeto ocorreram no laboratório de pesquisa localizado na sala 337 do campus Anglo. Esse ambiente foi escolhido

devido ao fluxo e ocupação por docentes e discentes durante o período letivo, o que proporcionou um cenário realista para os experimentos. Além disso, a infraestrutura e os recursos disponíveis no laboratório contribuíram significativamente para o desenvolvimento e a validação do sistema proposto. O laboratório possui um ponto de acesso a rede Internet independente da rede WUFPEL, que exige um login dos usuários que estabelecer a conexão e limita o tempo de conexão. Além disso, um PC é configurado e disponibilizado continuamente como servidor, executando o broker Mosquitto MQTT e também conectado localmente na rede Wi-fi existente na sala, o que permite a transferência e processamento eficiente de dados em ambiente controlado.

Após a definição e configuração da infraestrutura com o broker MQTT, este PC representa a partir de então o servidor de computação de borda, inicialmente definido no modelo apresentado na Figura 4. Deste modo, resta implementar os demais objetos sensores IoT responsáveis por adquirir os dados do sistema. Conforme definido anteriormente, os objetos sensores de aquisição usam o microcontrolador ESP32. Um protótipo é construído usando uma protoboard que permite a conexão da ESP32 aos sensores já definidos: luminosidade, distância, movimento e som. Além disso, é estabelecida uma conexão como um cliente do tipo *Publisher* com o broker MQTT. Para realizar essa conexão, a ESP32 se conecta à rede Wi-Fi local e utiliza a biblioteca *PubSubClient* para se conectar ao broker MQTT.

Como os módulos sensores usados disponibilizam os dados por meio de um sinal analógico contínuo no tempo, é necessário usar as portas analógicas da ESP32. Estas portas são ligadas diretamente a conversores analógico-digital internos à ESP32 e disponibilizam os dados de modo discretizado digitalmente. Para alguns sensores, como por exemplo o sensor o sensor ultrassônico, existe disponível uma biblioteca específica como a *Ultrasonic*, que permite medir distâncias com base no tempo que um pulso sonoro leva para viajar até um objeto e retornar. Com esta biblioteca, a ESP32 pode controlar o sensor ultrassônico e calcular a distância em centímetros.

Uma consideração importante deve ser destacada em relação à leitura do sensor de som. Devido a natureza da grandeza medida, o som ambiente pode variar rapidamente e ter frequências que variam dentro da faixa audível (20 a 20 kHz). Por este motivo é necessário capturar várias amostras ao longo tempo e continuamente. Com este propósito, optou-se por criar um laço de leituras do sensor de som a fim de determinar amostras do som ambiente em um curto intervalo de tempo. Com o laço, ao acumular várias leituras do sensor de som durante um curto período, obtém-se uma média desses valores, isso resulta em uma leitura mais estável e representativa do nível médio de som durante o intervalo de tempo considerado.

A coleta de dados é então realizada através da leitura desses sensores. Os valores registrados por esses sensores são coletados e posteriormente publicados em um tópico MQTT denominado "337/values", formatados como um JSON utilizando a bibli-

oteca *ArduinoJson*. Esta biblioteca desempenha a função de formatação dos dados, permitindo que a ESP32 estruture as leituras dos sensores em um objeto JSON bem definido antes de sua publicação no tópico. As mensagens são publicadas a cada 1 segundo, garantindo uma taxa de amostragem consistente e regular.

Abaixo, na Figura 10, está um exemplo de mensagem que a ESP32 publica nesse tópico:


```
{
    'lightSensor': 2485,
    'ultrasonicSensor': 130,
    'pirSensor': 0,
    'soundSensor': 63
}
```


Figura 10 – Exemplo de mensagem formatada como JSON com dados dos sensores.

3.4 Configuração do Cliente Subscriber para recebimento e armazenamento dos dados

Esta seção descreve a configuração do Subscriber, que é um cliente responsável por se conectar ao servidor, receber os dados dos sensores e armazenar esses dados recebidos em um arquivo *.txt* para uma análise posterior. Embora nos testes ele tenha sido executado na mesma máquina que o servidor, é importante ressaltar que ele também pode ser executado em uma máquina diferente, se necessário.

Para implementar o Subscriber, é utilizada a linguagem de programação Python e a biblioteca *paho MQTT*, que oferece suporte para a comunicação MQTT. Conforme mostrado na Figura 11, o código ilustra a implementação detalhada deste componente.

```python
import paho.mqtt.client as mqtt
import json
import datetime

mqtt_broker = ''        # Endereço IP do servidor MQTT
mqtt_port = 1883        # Número da porta do servidor MQTT
topic = '337/values'    # Tópico para assinar

# Salva os dados dos sensores em um arquivo
def saveToArqValues(sensor_data):
    current_datetime = datetime.datetime.now()
    with open("nome_arquivo.txt", "a") as file:
        file.write(
            str(current_datetime) + " "
            + str(sensor_data["lightSensor"]) + " "
            + str(sensor_data["ultrasonicSensor"]) + " "
            + str(sensor_data["pirSensor"]) + " "
            + str(sensor_data["soundSensor"]) + "\n"
        )

# Callback chamada quando se conecta com o broker MQTT
def on_connect(client, userdata, flags, rc):
    print("Conectado ao broker MQTT com código de resultado:", rc)
    client.subscribe(topic)

# Callback chamada quando uma mensagem é recebida no tópico
def on_message(client, userdata, msg):
    sensor_data = json.loads(msg.payload.decode('utf-8'))
    saveToArqValues(sensor_data)

# Configuração do cliente MQTT
client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message

client.connect(mqtt_broker, mqtt_port)  # Conexão ao broker MQTT

client.loop_forever()   # Loop de comunicação com o broker MQTT
```

Figura 11 – Código em Python do subscriber

Inicialmente são importadas as bibliotecas necessárias para implementação em Python como mostrado no Algoritmo **??**, incluindo a biblioteca *paho.mqtt.client* para comunicação MQTT e a biblioteca *json* para processamento de dados no formato JSON, além da biblioteca *datetime* para lidar e armazenar com informações de data e hora. A seguir, são apresentadas informações para estabelecer a conexão com o broker MQTT. Isso inclui a especificação do endereço IP, da porta do servidor MQTT usado para comunicação e do tópico em que o cliente em questão vai se conectar para receber as mensagens.

A função *SaveToArqValues* é responsável por salvar os dados dos sensores em um arquivo .txt, onde esses dados são registrados com informações de data/hora para fins de monitoramento e análise posterior. Abaixo, na Figura 12 segue um exemplo de como esses dados são armazenados no arquivo, mantendo a seguinte ordem em

cada linha: data, horário, dados do sensor de luz, dados do sensor ultrassônico, dados do sensor de presença e dados do sensor de som.

Timestamp	Luz	Ultr.	PIR	Som
2023-08-04 13:56:18.960244	2485	219	4095	0
2023-08-04 13:56:20.475816	2481	221	0	2671
2023-08-04 13:56:22.007012	2484	221	0	0
2023-08-04 13:56:23.522583	2485	219	0	0
2023-08-04 13:56:25.058605	2484	221	4095	0
2023-08-04 13:56:26.574177	2449	221	0	0

Figura 12 – Dados do dia 04/08/2023 - Período específico. Fonte: Própria

A função *on_connect* é ativada quando uma conexão bem-sucedida é estabelecida com o broker MQTT. Neste ponto, o Subscriber assina o tópico MQTT definido para receber mensagens relacionadas aos dados do sensor. E a função *on_message* é acionada sempre que chega uma mensagem sobre um tópico assinado pelo Subscriber. Ela é responsável por extrair os dados do sensor contidos na mensagem recebida e salvá-los no arquivo mencionado anteriormente.

4 RESULTADOS OBTIDOS

Este capítulo apresenta os experimentos realizados antes, durante e depois do desenvolvimento do sensor virtual, para avaliação ele. Para isso são explorados diferentes cenários de uso e configurações para as variáveis globais propostas visando ajustar a precisão do sensor. Lembrando que o objetivo principal é termos um sistema capaz de informar em tempo real se a sala está ocupada ou não durante uma jornada de trabalho. Como o estudo de caso foi realizado em um laboratório de pesquisa, a rotina de uso da sala é diferente de uma sala de aula, que possui horários bem definidos para utilização. Os laboratórios de pesquisa são usados por alunos de iniciação científica, pós-graduação e docentes em horários complementares às aulas e demais compromissos, ou seja, um uso não uniforme, onde em um certo momento existe um número expressivo de pessoas e em outros poucas pessoas, normalmente sentadas sem realizar movimentos ou ainda, em locais fora da cobertura dos sensores.

Os experimentos são realizados de modo a avaliar o comportamento do sistema proposto em diferentes contextos. Inicialmente avalia-se a importância da variável luminosidade na sala, uma das mais importantes no seu monitoramento, visto que normalmente durante a ocupação da sala a primeira ação realizada é ligar as luzes. Entretanto, algumas salas do Campus Anglo são bem iluminadas e durante um dia com boa luminosidade as luzes podem não ser ligadas, o que configura um contexto para avaliar o comportamento do sistema. Deste modo exploram-se dois cenários: (1) com a luz da sala ligada e (2) com a luz da sala desligada.

4.1 Análise dos sensores: dados produzidos

Nesta seção são apresentadas as análises dos dados coletados em cada sensor durante os experimentos iniciais. Essas análises foram realizadas para identificar tendências, padrões e informações relevantes que podem fornecer percepções valiosas para o desenvolvimento do sensor virtual. Para a coleta dos dados, inicialmente foi definido a posição P1 da Figura 13 como um local estratégico dentro do laboratório, para direcionar o sensor ultrassônico de modo a monitorar o corredor, detectando a

presença de alguém que passe por ele. Enquanto o sensor PIR é direcionado para a área da sala com maior movimentação de pessoas.

Além dessa posição, também foi definida uma segunda posição de testes, denominada P2, conforme demonstrado na Figura 13. Essa posição está estrategicamente localizada nas proximidades da porta, com o sensor ultrassônico direcionado em sua direção. Essa configuração não apenas possibilita a detecção de movimentos de pessoas que passam pelo corredor, mas também permite a supervisão do estado da porta, permitindo verificar se está aberta ou fechada.

As duas posições foram escolhidas para os testes porque estão localizadas próximas ao corredor da sala, onde há maior fluxo de pessoas no ambiente.

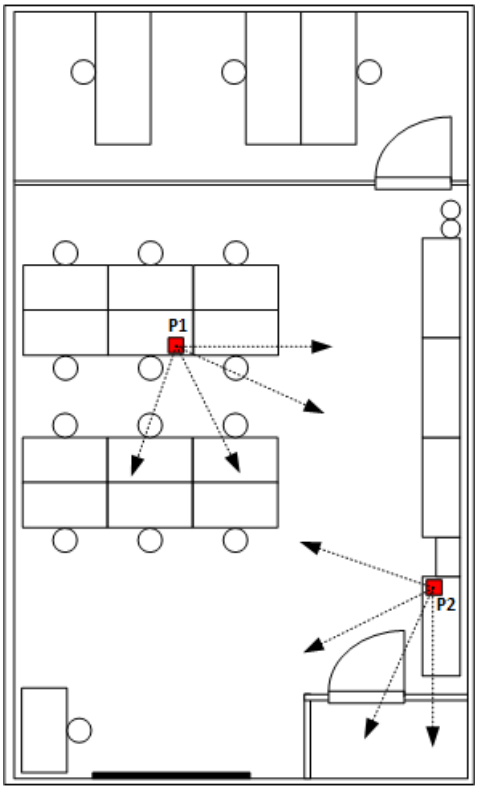

Figura 13 – Esboço do leiaute da sala 337 e posicionamento do sistema de monitoramento. Fonte: Própria

Os dados obtidos na posição P1 são apresentados em gráficos nas Figuras 14, 15 e 16.

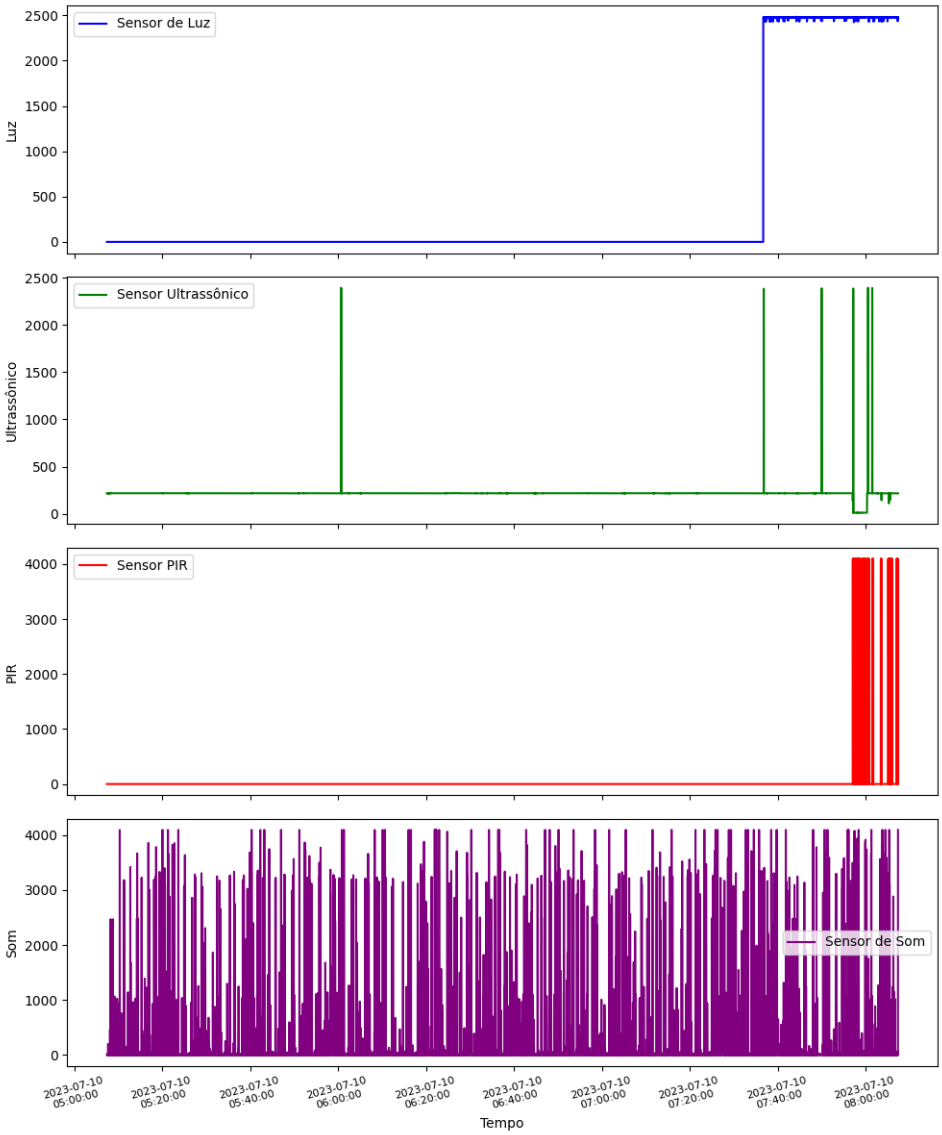

Figura 14 – Dados do dia 10/07/2023-1. Fonte: Própria

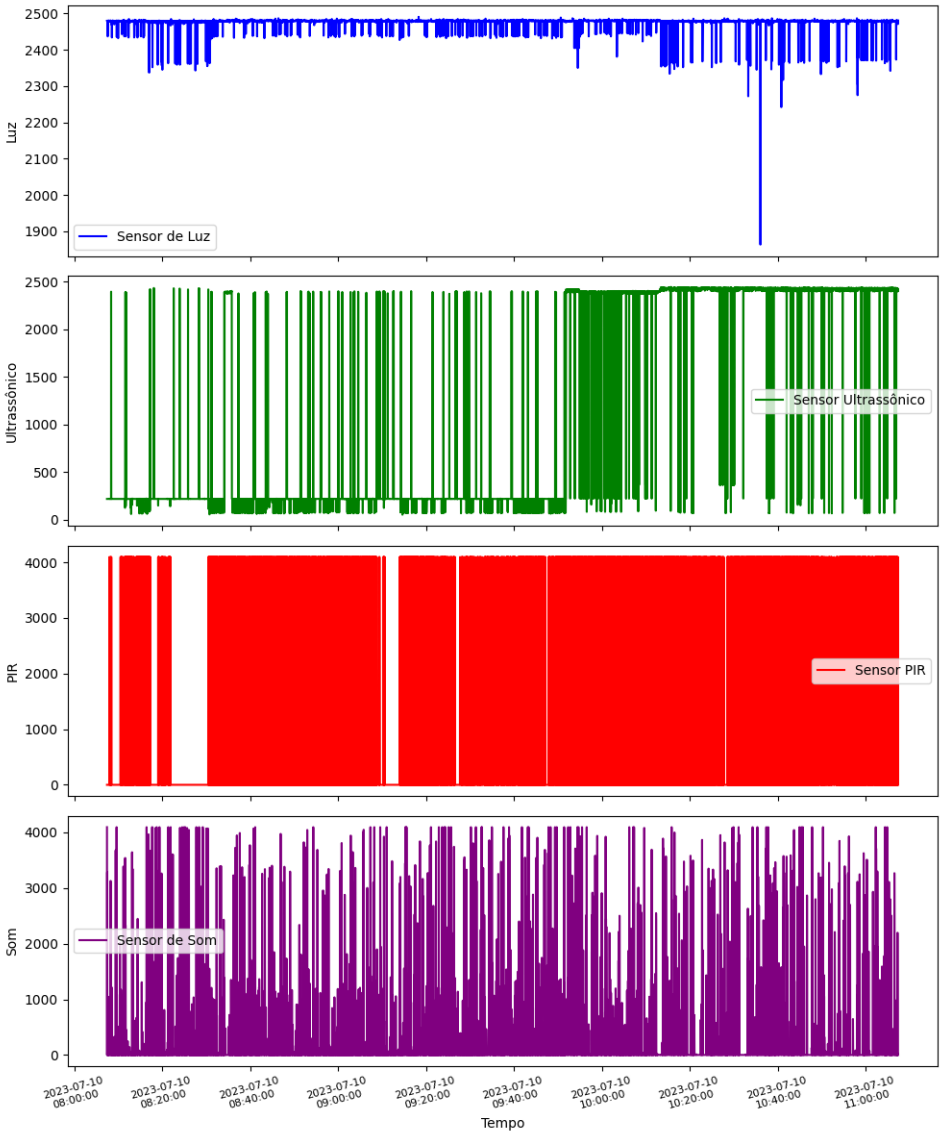

Figura 15 – Dados do dia 10/07/2023-2. Fonte: Própria

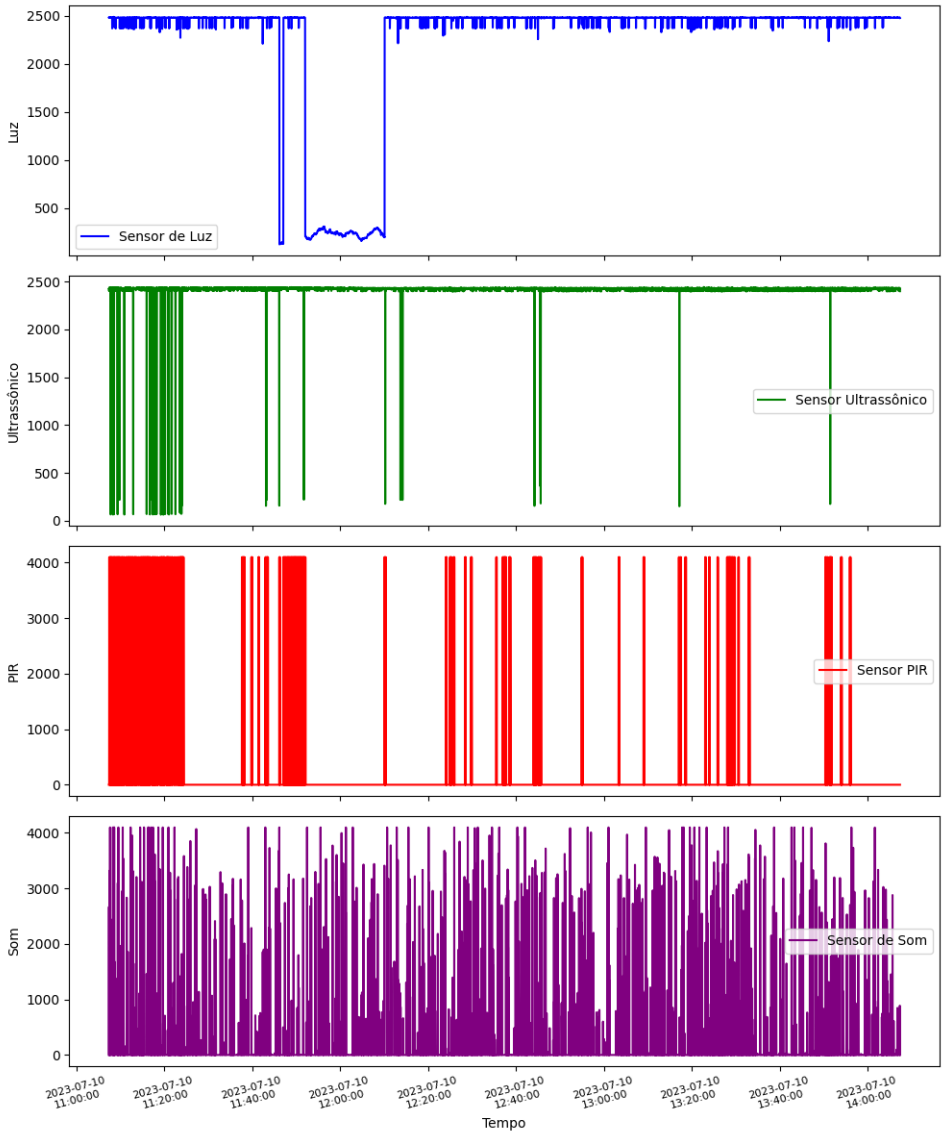

Figura 16 – Dados do dia 10/07/2023-3. Fonte: Própria

Ao analisar os gráficos, a primeira observação relevante diz respeito aos ruídos nos sensores, tanto no sensor ultrassônico quanto no sensor de som. No caso do sensor ultrassônico, analisando o gráfico da Figura 14, nota-se que ele mantém um valor constante entre 0 e 500, indicando que detecta uma parede ou objeto a uma distância de 0 a 500cm. No entanto, em alguns momentos, esse valor apresenta picos significativos, chegando a ultrapassar os 2000cm, ocasionados por problemas na fixação e posicionamento do sensor, gerando ruidos indesejados.

Apesar do ruído presente, as informações úteis desse sensor são reveladas quando ele registra valores significativamente inferiores em comparação às leituras anteriores. Isso se torna evidente ao analisar os dados registrados no arquivo .txt. Por exemplo:

Timestamp	Luz	Ultr.	PIR	Som
2023-07-10 08:05:16.120017	2479	217	0	1266
2023-07-10 08:05:17.260685	2480	217	0	0
2023-07-10 08:05:18.385650	2477	110	0	0
2023-07-10 08:05:19.495012	2480	217	4095	0
2023-07-10 08:05:20.635596	2480	217	4095	643

Figura 17 – Dados do dia 10/07/2023 - Período específico. Fonte: Própria

Na Figura 17, que representa um período específico contido no gráfico da Figura 14, é possível observar que os valores registrados pelo sensor ultrassônico permanecem constantes em 217, depois diminuem para 110 e, em seguida, retornam a 217. Essa sequência de leituras indica que o sensor detectou a presença de alguém na sala. Além disso, é possível observar uma redundância por meio do sensor PIR, que estava continuamente marcando 0 e, em seguida, saltou para 4095. Com base nesses dados, podemos concluir que há presença de alguém no ambiente.

Já nos gráficos das Figuras 15 e 16, é perceptível que a presença de ruído no sensor ultrassônico é mais significativa. No desenvolvimento do sensor virtual, é importante ignorar esses ruídos, pois eles podem afetar a precisão das medições e a interpretação dos dados.

Para os dados obtidos pelo sensor de som, observa-se que o ruído é ainda mais visível e contínuo. Este ruído se manifesta de maneira notável, mesmo em períodos em que não há presença de ninguém no ambiente, como por exemplo entre os horários da meia-noite às 6h. Até mesmo em cenários onde é provável que haja alguém no ambiente, o sensor de som enfrenta desafios para distinguir de maneira precisa entre ruídos intrínsecos ao ambiente e os sons gerados por pessoas no ambiente. Essa dificuldade em fazer essa distinção afeta a confiabilidade das medições e a eficiência do sensor na detecção de eventos de interesse.

Para tentar reduzir este problema, foram realizadas diversas tentativas de ajustes no processo de aquisição de dados desse sensor, com a expectativa de alcançar resultados e dados com menor nível de ruído.

Os gráficos das Figuras 14, 15 e 16 representam uma configuração de leitura do sensor de som que realiza um ciclo de leitura com 64 iterações, onde ele coleta dados em cada iteração e, em seguida, calcula a média desses valores. Outra configuração testada envolveu a introdução de um atraso de 10 milissegundos entre cada iteração desse laço, conforme ilustrado nos gráficos das Figuras 18 e 19. Por fim, a Figura 20 representa os testes do sensor de som realizando 32 iterações sem a inclusão do atraso.

Resumindo, após essa série de testes com diferentes configurações de leitura do

sensor de som, ficou evidente que o problema do ruído persistiu de forma significativa e prejudicial. Infelizmente, os dados obtidos deste sensor não puderam ser utilizados de maneira eficaz no desenvolvimento do sensor virtual devido à constante presença de ruído.

Devido a esses desafios encontrados, esses resultados destacaram a falta de confiabilidade deste sensor em ambientes ruidosos. Como sugestão para trabalhos futuros, é considerável a aplicação de um filtro de ruído nos dados coletados por este sensor. Essa abordagem pode representar uma solução viável para melhorar a qualidade das medições, tornando-as mais precisas e aumentando assim a eficácia do sensor virtual.

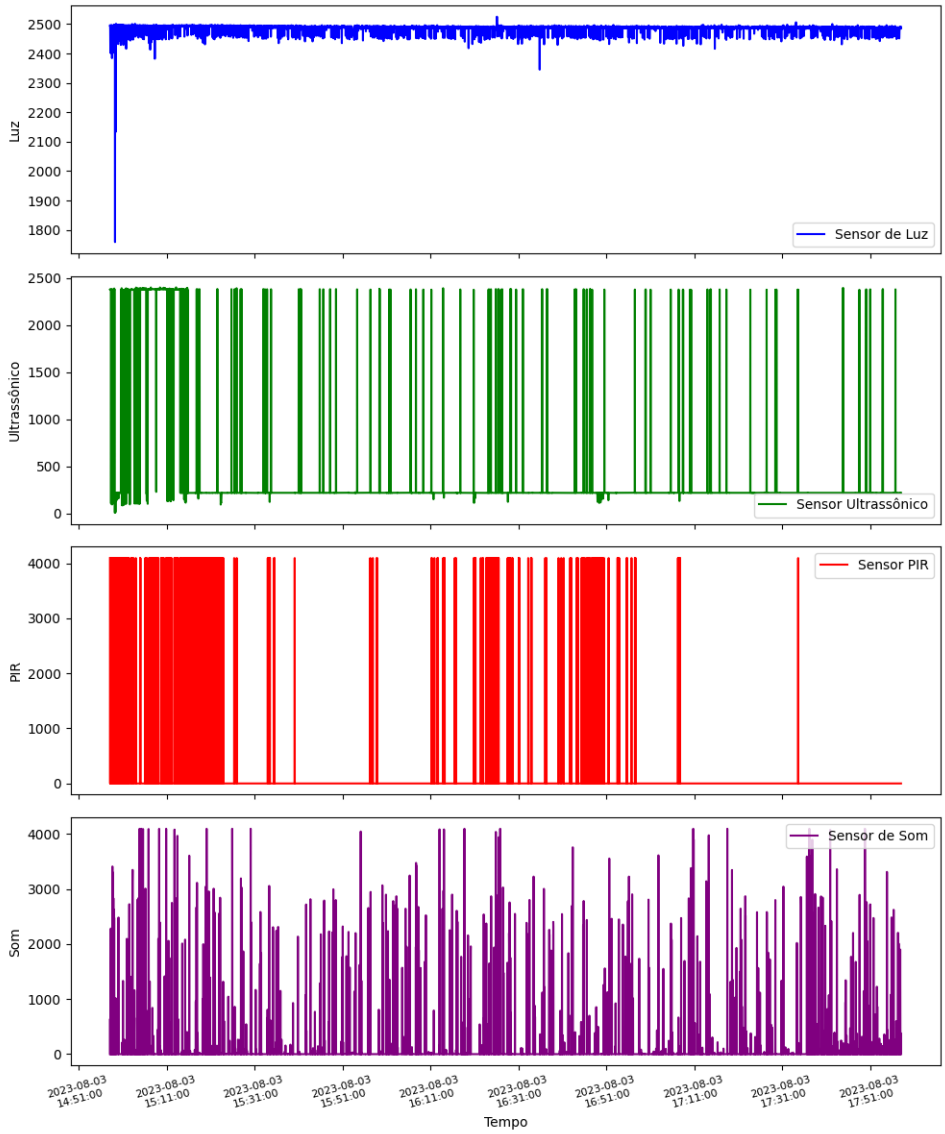

Figura 18 – Dados do dia 03/08/2023-1. Fonte: Própria

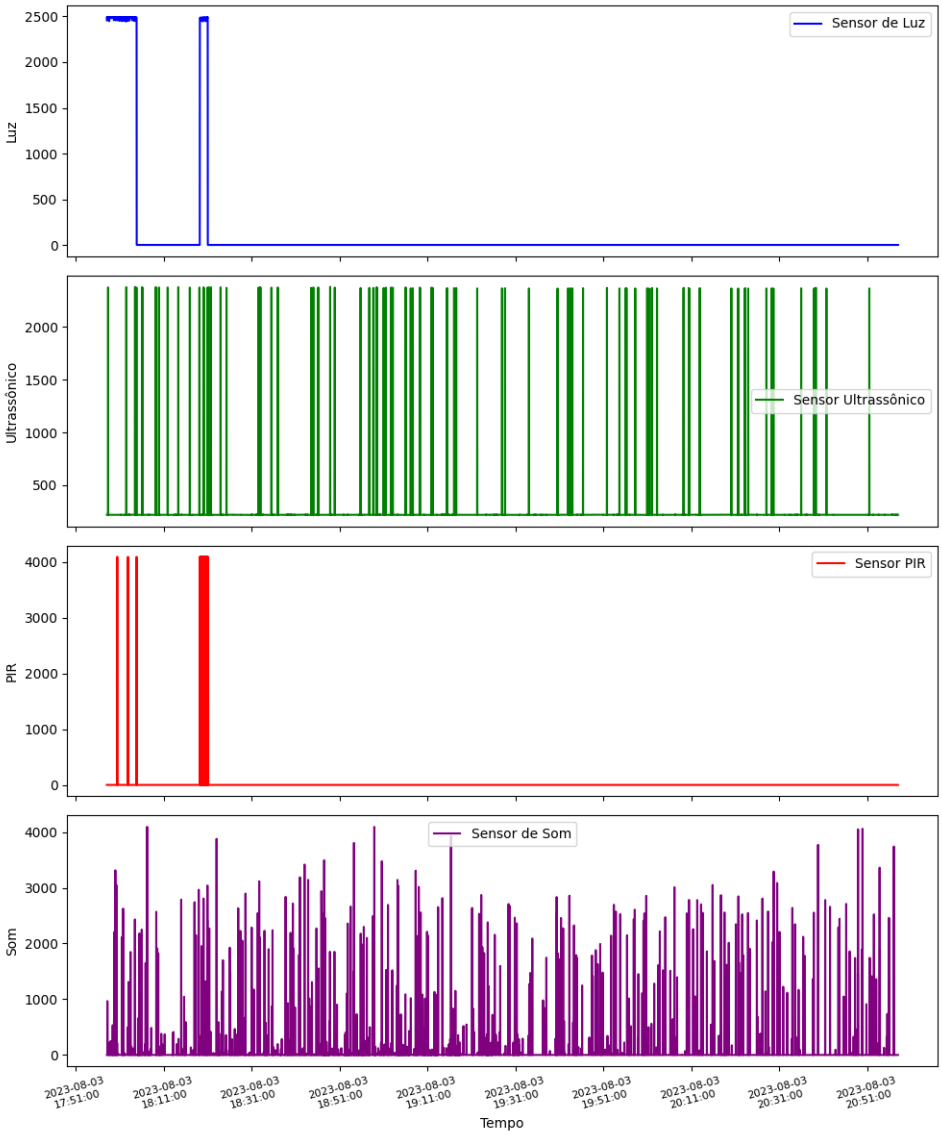

Figura 19 – Dados do dia 03/08/2023-2. Fonte: Própria

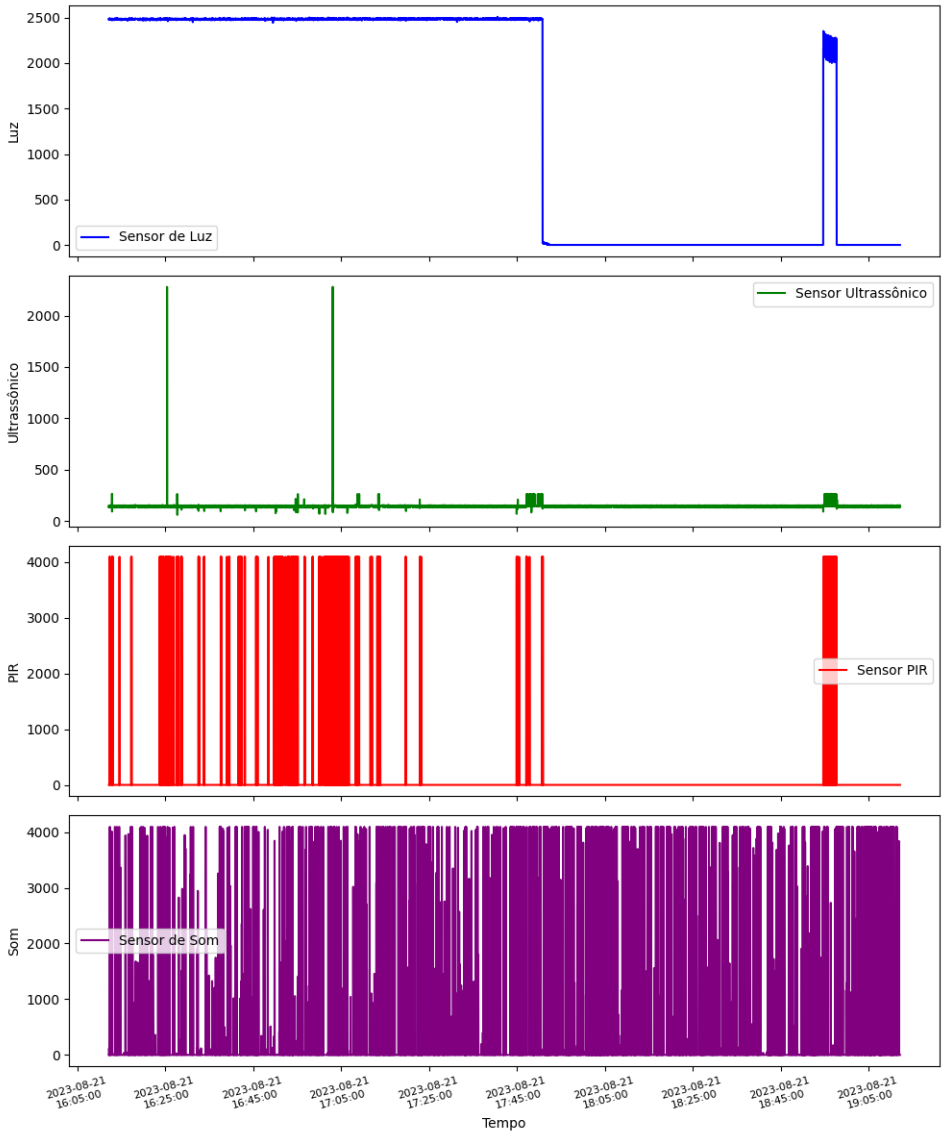

Figura 20 – Dados do dia 21/08/2023-1. Fonte: Própria

Apesar dos desafios enfrentados com esses sensores, ao analisar os outros dois sensores, ou seja, o sensor de luz e o sensor PIR, torna-se evidente que eles fornecem uma base sólida e confiável para o desenvolvimento do sensor virtual. Estes dois sensores desempenham um papel fundamental ao oferecer informações consistentes e valiosas, contribuindo significativamente para o desenvolvimento e a eficácia do sensor virtual.

Referente ao sensor de luz, pode-se inferir que valores superiores a 1000 indicam uma iluminação adequada no ambiente para a presença de pessoas. Da mesma forma, o sensor PIR, com leituras acima de 4000, indica a presença de movimentação de pessoas no ambiente.

Para compreender o que o sensor virtual deve indicar, podemos analisar o gráfico da Figura 16 ao meio-dia. Neste momento, não há mais presença de pessoas no ambiente, conforme demonstram os dados dos três sensores que serão considerados.

Além disso, ao examinar o gráfico da Figura 19, observa-se que entre 18h11 e 18h31, houve um período de presença de pessoas, como indicado pelos sensores de luz e PIR.

O gráfico representado na Figura 20, assim como todos os demais gráficos com data a partir do dia 21/08/2023, foram obtidos utilizando a configuração dos sensores localizados na posição P2. Ao analisar esse gráfico, torna-se evidente que, além dos períodos em que os valores registrados pelo sensor ultrassônico são menores, como discutido previamente, existem momentos em que o sensor apresenta leituras mais elevadas (com exceção dos valores acima de 2000cm, que são considerados ruído). Isso sugere que, em determinados instantes, o sensor estava detectando a porta, indicando uma distância relativamente próxima. Entretanto, posteriormente, as leituras do sensor aumentaram significativamente, o que pode ser interpretado como o sensor passando a detectar a parede localizada atrás da porta, resultando em leituras mais distantes.

Portanto, o posicionamento do sistema na posição P2 ao invés da posição P1, contribuiu para uma análise mais completa e precisa do comportamento do sensor ultrassônico, proporcionando uma representação mais informativa dos dados coletados e ajudando a captar melhor a movimentação de pessoas no ambiente.

4.2 Desenvolvimento do Sensor Virtual

Esta seção apresenta o desenvolvimento do sensor virtual com base nas análises dos dados obtidos pelos sensores, conforme discutido anteriormente. O objetivo principal do sensor virtual é fornecer informações sobre a presença de pessoas no ambiente, levando em consideração os dados dos sensores de luz, PIR e ultrassônico.

O sensor virtual proposto funciona com base em limiares(*thresholds*) específicos estabelecidos para cada um dos sensores utilizados:

- **Sensor de Luz:** Um limiar de 1000 é definido para este sensor. Quando a luminosidade ambiente ultrapassa esse valor, considera-se que a luz está acesa no ambiente.
- **Sensor PIR:** O limiar estabelecido para este sensor é de 4000. Quando ele detecta valores acima desse limiar, considera-se que há presença no ambiente.
- **Sensor Ultrassônico:** Este sensor requer uma abordagem mais complexa. Calcula-se a média dos últimos 10 valores lidos por ele, e qualquer leitura que esteja dentro de uma margem de 10 centímetros abaixo da média é considerada indicativa de presença. Por exemplo, se a média calculada for 200 cm, qualquer

valor lido pelo sensor abaixo de 190cm será interpretado como presença. Além disso, para eliminar ruídos, valores acima de 2000cm são descartados.

A Figura 22 apresenta as variáveis globais e a função para o cálculo da média dinâmica dos dados do sensor ultrassônico.

O fluxograma da Figura 21 e o código presente na Figura 23 demonstram como funciona a implementação do sensor virtual. Para entender seu funcionamento, é importante analisar os diferentes cenários que podem ocorrer.

O código da implementação do sensor virtual faz uso de duas variáveis globais auxiliares, *lastPresenceTime* e *noise*, que desempenham papéis fundamentais em sua funcionalidade.

- *lastPresenceTime* armazena o momento da última detecção de presença pelo sistema. Essa informação é crucial para determinar o intervalo de tempo desde a última detecção.
- *noise* é uma variável booleana que nos informa se a última presença detectada quando a luz estava apagada foi resultado de ruído ou não. Essa distinção é vital para evitar falsas detecções.

4.2.1 Cenário 1: com luz ligada

O algoritmo começa por verificar se o sensor de luz registra um valor acima do limite predefinido *lightThreshold*. Isso indica que a luz no ambiente está acesa. Nesse cenário, o código realiza duas verificações adicionais:

1. **Presença Detectada:** Se o sensor ultrassônico indica que algo está a uma distância menor que a média da janela de amostragem (*ultrasonicThreshold* - 10) ou o sensor PIR detecta movimento acima do limite *pirThreshold*, o código considera que há alguém na sala. Ele registra o momento da última detecção de presença *lastPresenceTime* como o tempo atual e retorna "verdadeiro", indicando que a presença foi detectada.
2. **Ausência Prolongada:** Se o tempo desde a última detecção de presença for maior que 900 segundos (15 minutos), o código retorna "falso", indicando que a presença não foi detectada por um longo período de tempo.
3. **Presença Recente:** Se a última detecção de presença ocorreu há menos de 900 segundos, o código retorna "verdadeiro", indicando que a presença foi detectada recentemente.

4.2.2 Cenário 2: com luz desligada

Quando a luz está apagada, o algoritmo define uma série de verificações para distinguir entre presença real e possíveis ruídos:

1. **Presença Detectada**: Se o sensor ultrassônico indica presença (a distância é menor que *ultrasonicThreshold* - 10 ou o sensor PIR detecta movimento acima de *pirThreshold*), o código verifica se a última detecção de presença ocorreu há menos de 10 segundos. Se isso for verdade, considera que há alguém na sala, então atualiza *lastPresenceTime* e define *noise* como "falso", indicando que essa presença detectada não é ruído, e retorna "verdadeiro".

2. **Presença por Ruído**: Se a última detecção de presença ocorreu há mais de 10 segundos, o código considera que a presença detectada é ruido, então atualiza *lastPresenceTime* e define *noise* como "verdadeiro", indicando que é ruído. Retorna "falso".

3. **Presença Recente**: Se nenhuma das condições anteriores for atendida e o tempo desde a última detecção de presença for menor que 30 segundos (*lastPresenceTime* + 30 > *now()*) e não houver ruído (!*noise*), o código retorna "verdadeiro", indicando que a presença foi detectada a menos de 30 segundos e não é ruído.

4. **Ausência Prolongada**: Se nenhuma das condições anteriores for atendida, o código retorna "falso", indicando que a presença não foi detectada por pelo menos 30 segundos.

Essas verificações detalhadas permitem que o sensor virtual tome decisões precisas sobre a detecção de presença, levando em consideração diversos cenários e evitando falsas detecções devido a ruídos.

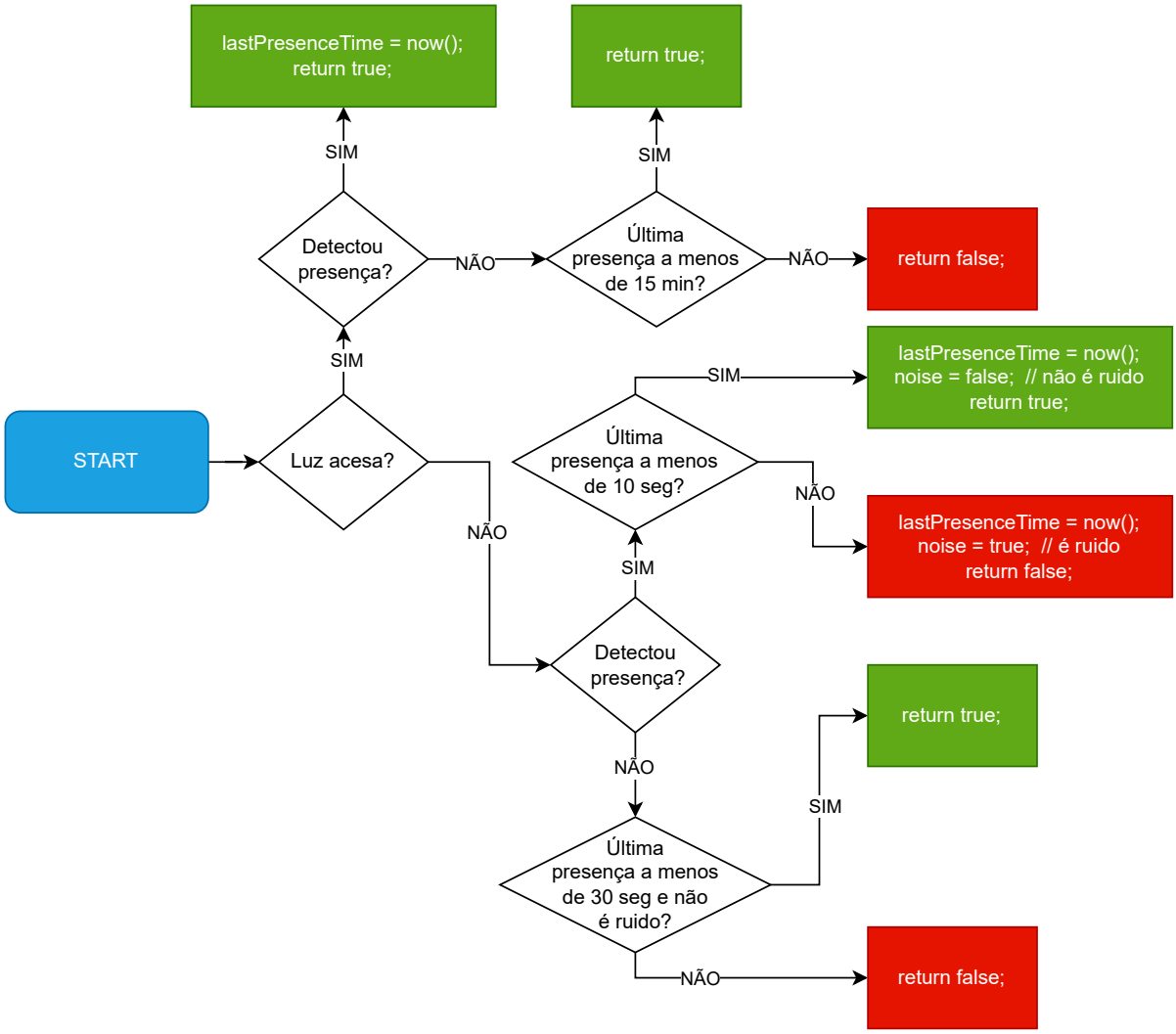

Figura 21 – Fluxograma do Sensor Virtual

```
const int ultrasonicWindowSize = 10;  // Tamanho da janela de amostragem para a média dinamica
int ultrasonicValues[ultrasonicWindowSize];
int ultrasonicValuesIndex = 0;

// Função que calcula o threshold do sensor ultrassonico
int calculateUltrasonicAverage() {
    int sum = 0;
    for (int i = 0; i < ultrasonicWindowSize; i++) {
        sum += ultrasonicValues[i];
    }

    return sum / ultrasonicWindowSize;
}
```

Figura 22 – Código em C++ das variáveis e função para o cálculo da média dinâmica do Sensor Ultrassônico

```
bool checkPresence(DynamicJsonDocument values) {
    // Define os thresholds para cada sensor
    const int lightThreshold = 1000;
    const int pirThreshold = 4000;
    const int soundThreshold = 100;
    // se está acima de 2000 é porque tem ruido
    if (values["ultrasonicSensor"] > 2000) {
        values["ultrasonicSensor"] = ultrasonicValues[ultrasonicValuesIndex - 1];
// pegando o valor anterior do sensor (valor sem ruido)
    }
    int ultrasonicThreshold = calculateUltrasonicAverage();

    if (values["lightSensor"] > lightThreshold) { // Luz está acesa
        if (values["ultrasonicSensor"] < ultrasonicThreshold - 10 || values["
pirSensor"] > pirThreshold) {
            lastPresenceTime = now();
            return true; // Há alguém na sala
        }
        else if (lastPresenceTime + 900 < now()) {
            return false; // Não detectou presença a pelo menos 900 segundos
        }
        else {
            return true; // Detectou presença a pelo menos 900 segundos
        }
    }
    else if (values["ultrasonicSensor"] < ultrasonicThreshold - 10 || values["
pirSensor"] > pirThreshold) { // Luz está apagada
        if (lastPresenceTime + 10 > now()) {
            lastPresenceTime = now();
            noise = false; // não é ruido
            return true;   // Há alguém na sala
        }
        lastPresenceTime = now();
        noise = true; // É ruido
        return false; // Presença que detectou provavelmente é ruido
    }
    else if (lastPresenceTime + 30 > now() && !noise) {
        return true; // Detectou presença a menos de 30 seg que não é ruido
    }
    else {
        return false;
    } // Não detectou presença a pelo menos 30 seg

    // Armazena o novo valor do sensor ultrassonico na janela de amostragem
    ultrasonicValues[ultrasonicValuesIndex] = newValue;
    ultrasonicValuesIndex = (ultrasonicValuesIndex + 1) % ultrasonicWindowSize;
}
```

Figura 23 – Código em C++ do Sensor Virtual

A função *checkPresence* é chamada no código para verificar a presença com base nas condições definidas, e ela recebe os dados dos sensores como parâmetros. Essa função é invocada na função *loop*. A função *loop* é um laço infinito que executa continuamente o código contido dentro dela.

Em seguida, com base no valor retornado por essa função, a ESP32 publica "1"a cada 10 segundos caso a função retorne verdadeiro, e "0"se a função retorne falso,

em um novo tópico no servidor *MQTT*, chamado de "337/presence".

Posteriormente, outro cliente do tipo "subscriber"se conecta ao servidor *MQTT* e se inscreve nesse tópico para receber e armazenar as informações geradas pelo sensor virtual.

4.3 Cenário 1: com luz ligada

Neste cenário são realizados testes com diferentes valores especificamente para a variável *lastPresenceTime*. Nesse contexto, o sensor virtual considera períodos de "presença recente"com os valores de 30, 120 e 900 segundos atribuídos a essa variável. Ou seja, nos primeiros testes do sensor virtual é atribuido a variável *lastPresenceTime* o valor 30, indicando que o sensor virtual deixaria de indicar a presença de alguém no ambiente após 30 segundos sem detectar qualquer presença. O comportamento obtido pode ser observado nos gráficos das Figuras 24 e 25.

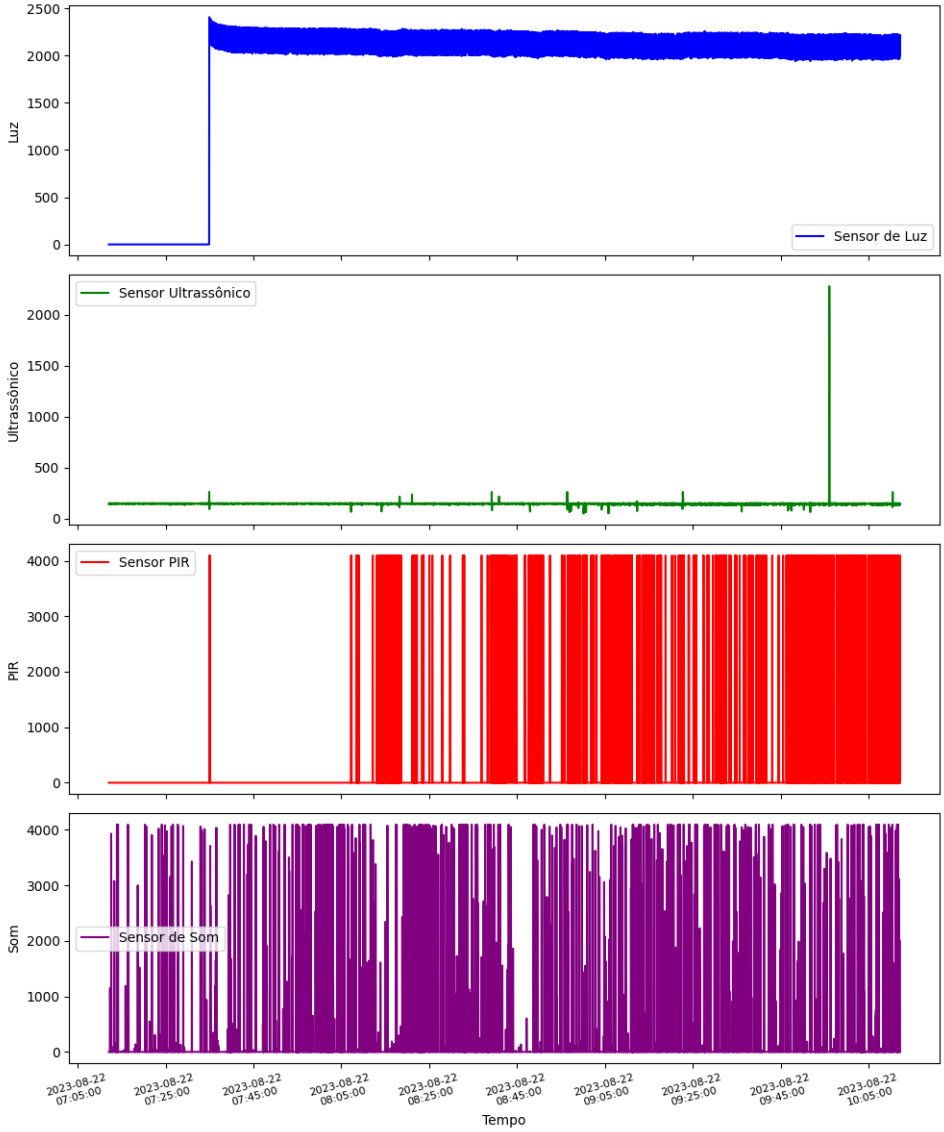

Figura 24 – Dados do dia 22/08/2023-1. Fonte: Própria

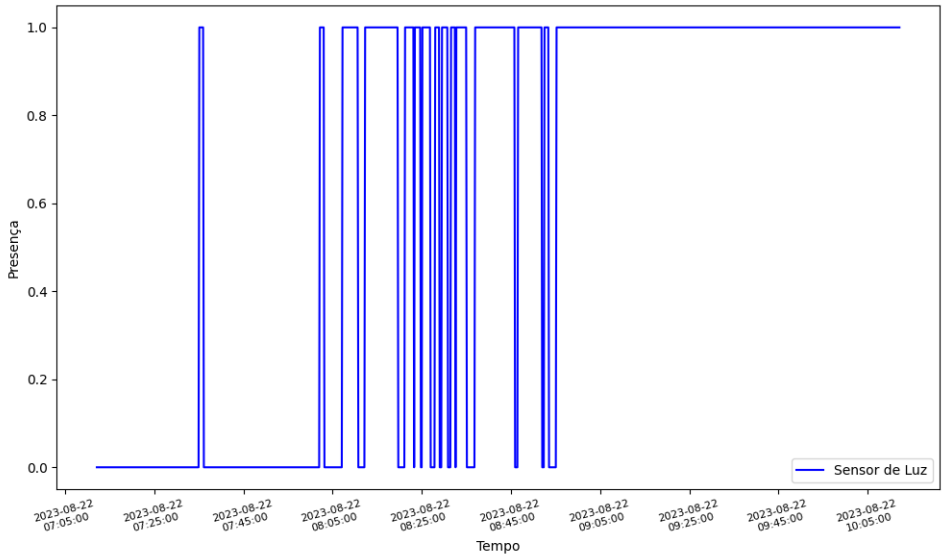

Figura 25 – Dados de teste do Sensor Virtual com 30s do dia 22/08/2023-1. Fonte: Própria

De modo a avaliar o comportamento do sensor virtual proposto é atribuido o valor de 120 segundos para a variável *lastPresenceTime*. Os resultados obtidos com este ajuste podem ser observados nos gráficos representados pelas Figuras 26 e 27.

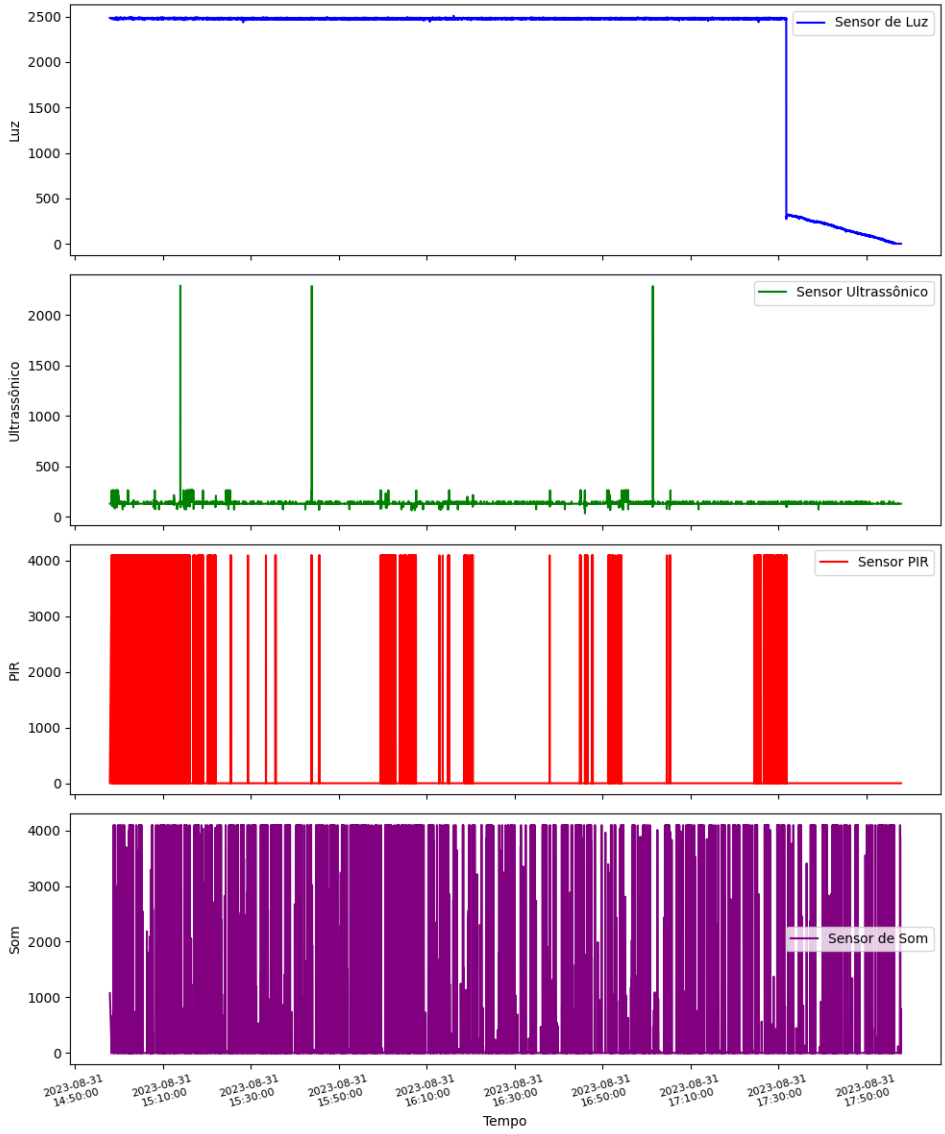

Figura 26 – Dados do dia 31/08/2023-1. Fonte: Própria

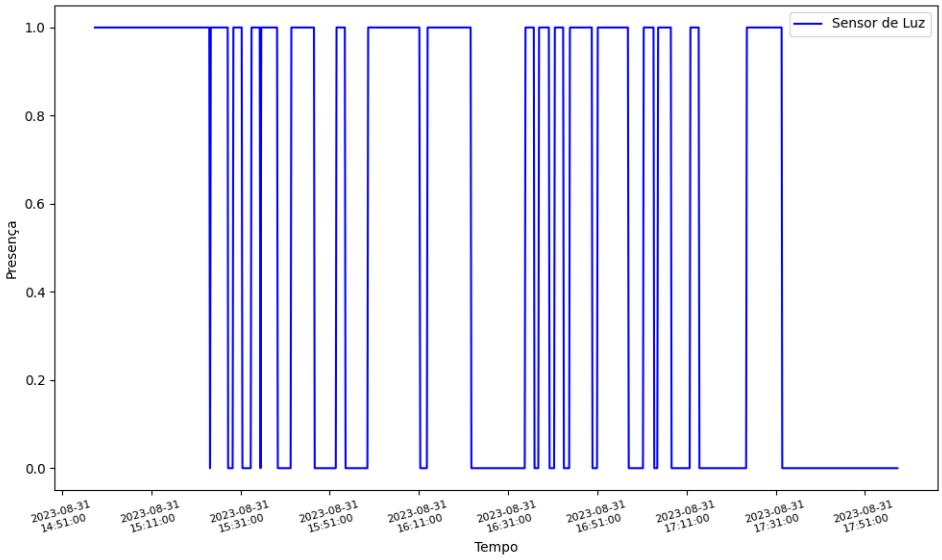

Figura 27 – Dados de teste do Sensor Virtual com 120s do dia 31/08/2023-1. Fonte: Própria

No gráfico da Figura 25, é possível observar que o sistema é capaz de detectar a presença, embora apresente uma considerável variação nos resultados. No entanto, ao analisar o gráfico da Figura 27, nota-se uma redução significativa dessa variabilidade. É importante salientar que o propósito do sensor virtual não é detectar o exato momento que houve ou não presença no ambiente, mas sim informar os períodos do dia em que houve presença. Portanto, a escolha de um intervalo de 900 segundos após a detecção de presença foi adotada para assegurar uma visão mais abrangente desses períodos de atividade.

Já para valores de 900 segundos (15 minutos), é possível observar os resultados nos gráficos representados pelas Figuras 28, 29, 30, 31, 32 e 33.

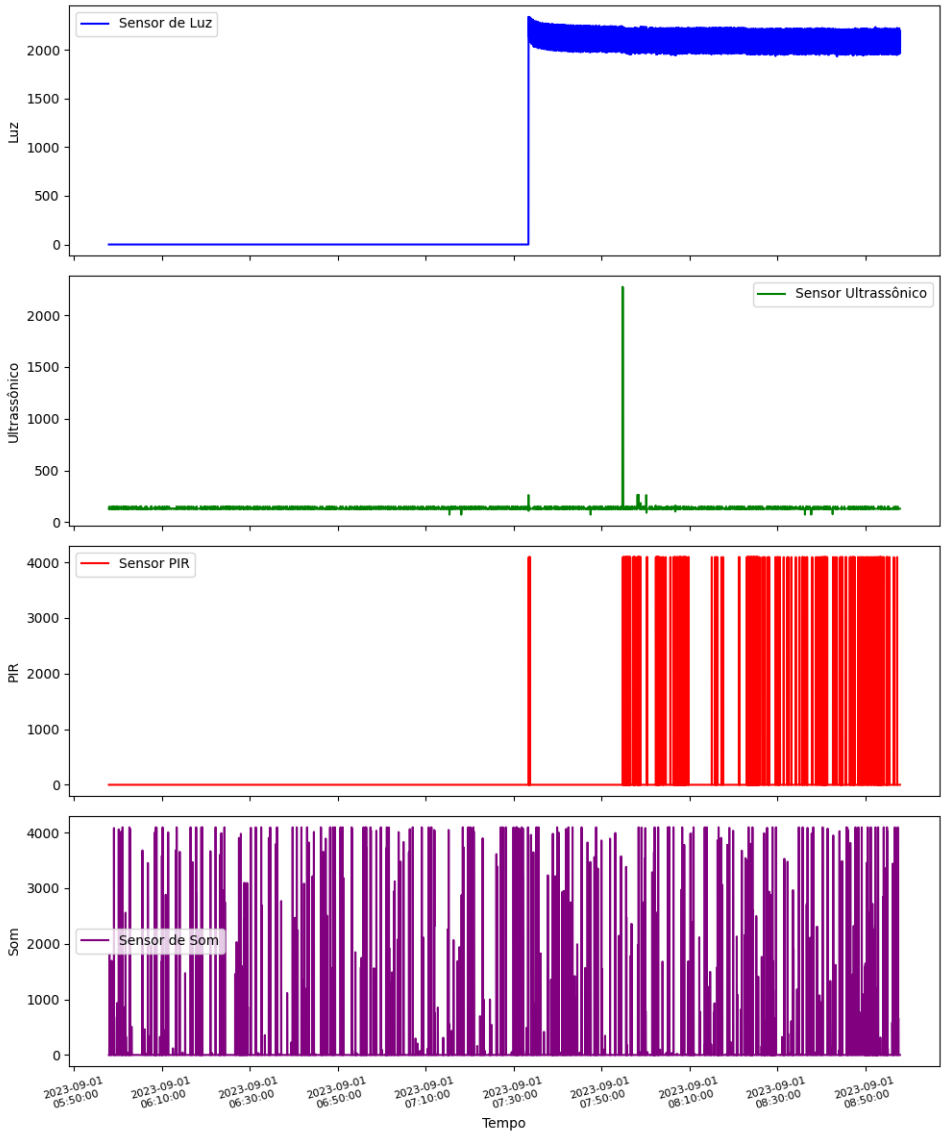

Figura 28 – Dados do dia 01/09/2023-1. Fonte: Própria

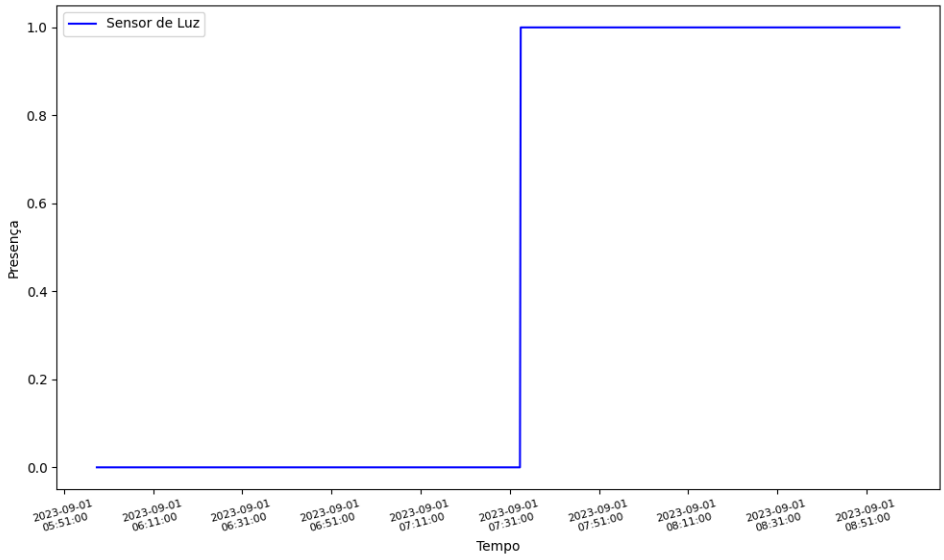

Figura 29 – Dados de teste do Sensor Virtual com 900s do dia 01/09/2023-1. Fonte: Própria

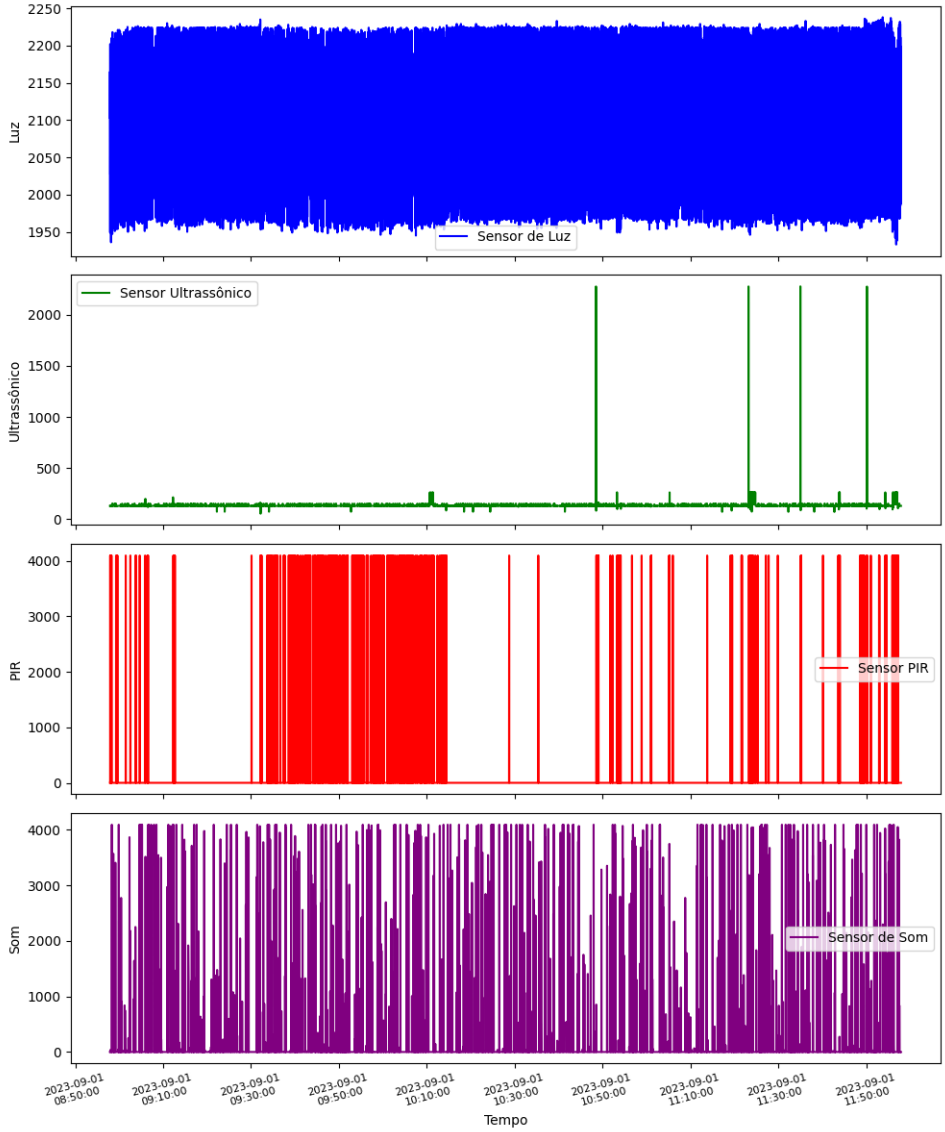

Figura 30 – Dados do dia 01/09/2023-2. Fonte: Própria

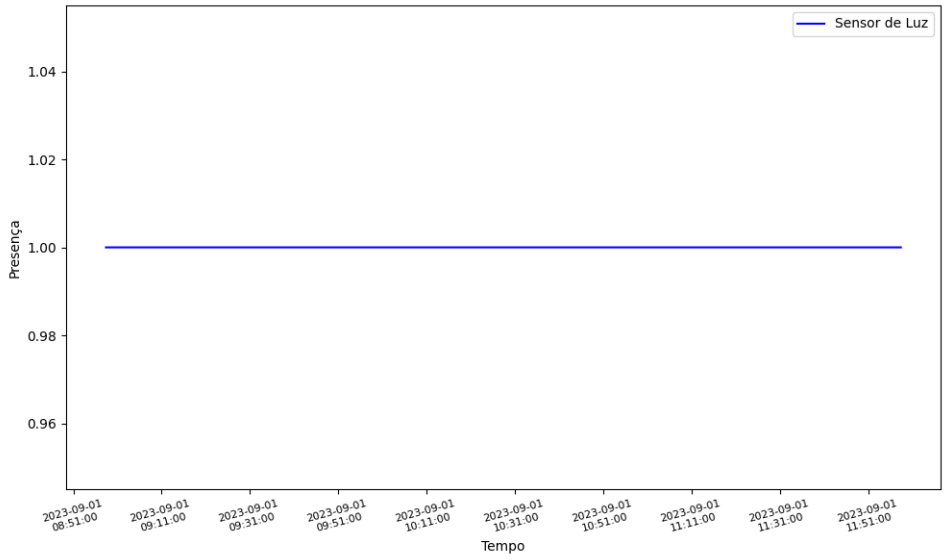

Figura 31 – Dados de teste do Sensor Virtual com 900s do dia 01/09/2023-2. Fonte: Própria

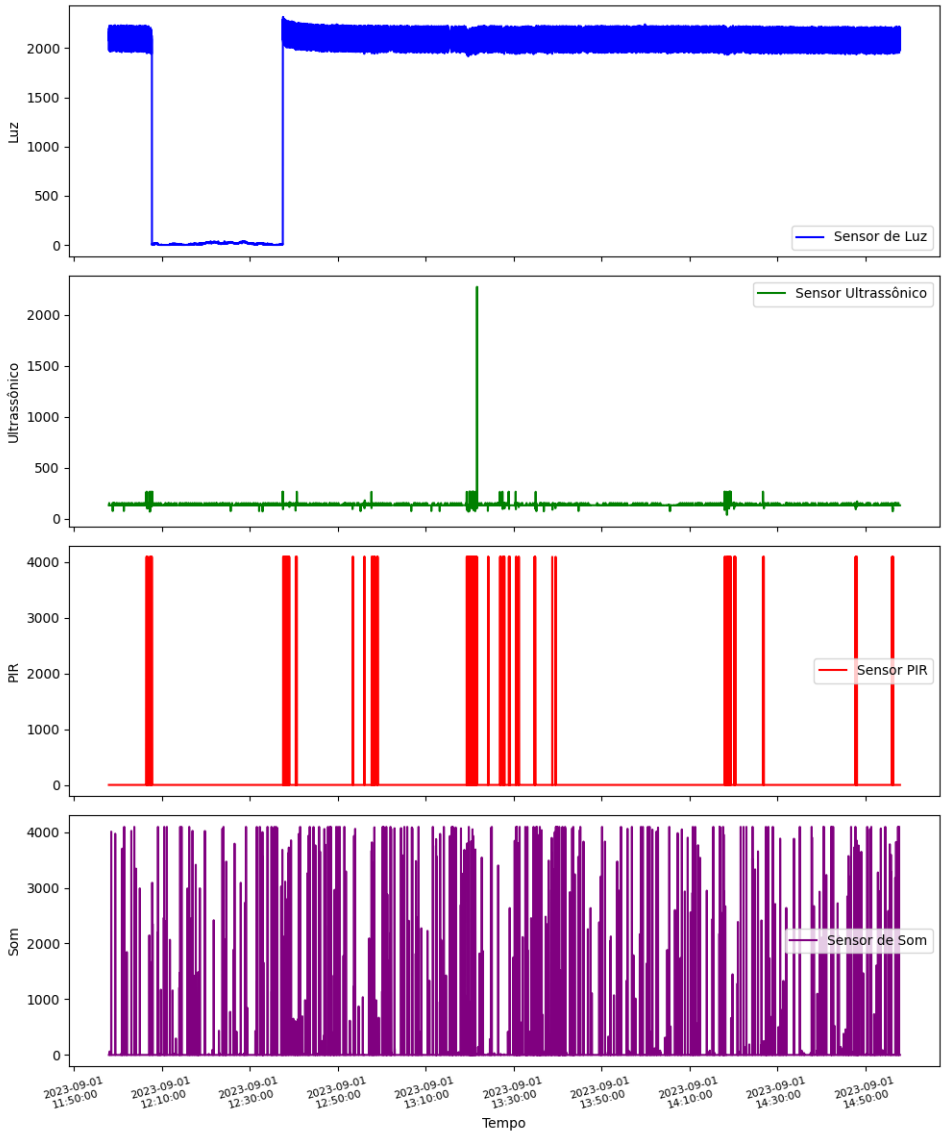

Figura 32 – Dados do dia 01/09/2023-3. Fonte: Própria

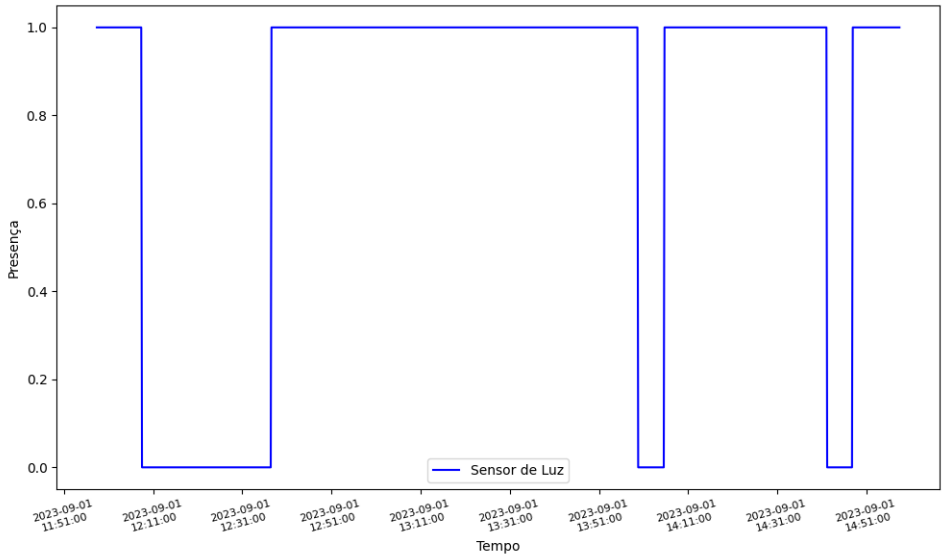

Figura 33 – Dados de teste do Sensor Virtual com 900s do dia 01/09/2023-3. Fonte: Própria

Pelos gráficos apresentados, pode-se observar que as variações dos dados nos gráficos do sensor virtual são significativamente menores, e em alguns casos, até inexistente como mostrado gráfico da Figura 31, em comparação com os gráficos dos primeiros testes do sensor virtual com 30s ou 120s conforme ilustrados nos gráficos das Figuras 25 e 27.

A utilização de um atraso de 900 segundos após a detecção de presença com a luz acesa se mostrou eficiente. Isso permite identificar claramente os momentos do dia em que a sala está ocupada e quando não está, como evidenciado no gráfico da Figura 29.

Entretanto, em situações em que a luz está acesa e o sistema não detecta presença por pelo menos 900 segundos, não se pode ter certeza se a sala está realmente desocupada ou se o sensor virtual não conseguiu identificar nenhum movimento. Isso pode ser observado no gráfico da Figura 33, no qual há três momentos em que o sensor virtual indica a ausência de presença na sala. O primeiro período ocorre das 12h08 às 12h37, onde de fato não houve presença na sala, conforme evidenciado ao analisar o gráfico da Figura 32, que essa ausência de presença foi registrada por todos os três sensores utilizados.

Os outros dois periodos foram das 13h59 às 14h05 e das 14h42 às 14h47, que, como dito anteriormente, durante esses intervalos de tempo, o sensor virtual indicou a ausência de presença na sala, mas não se pode descartar a possibilidade de que alguém possa ter estado na sala, mas o sensor não tenha detectado esse movimento por algum motivo.

Portanto, ao interpretar os dados do sensor virtual, é fundamental estar atento a possíveis nuances, principalmente em situações em que a luz está acesa e o sistema não detecta presença por pelo menos 900 segundos.

4.4 Cenário 2: com luz desligada

Neste caso, buscou-se investigar o comportamento do sensor virtual proposto em um cenário onde uma das principais variáveis do ambiente está desligada, ou seja, uma pessoa ou grupo de pessoas pode ter ocupado a sala sem ligar a luz do ambiente. O comportamento obtido pode ser analisado pelos gráficos das Figuras 34 e 35, onde observa-se que o sensor virtual também foi capaz de detectar presença mesmo com a luz apagada. No gráfico da Figura 34, é possível observar que os sensores físicos detectaram presença mesmo com a luz apagada, e o mesmo padrão é evidenciado no sensor virtual, como indicado no gráfico da Figura 35.

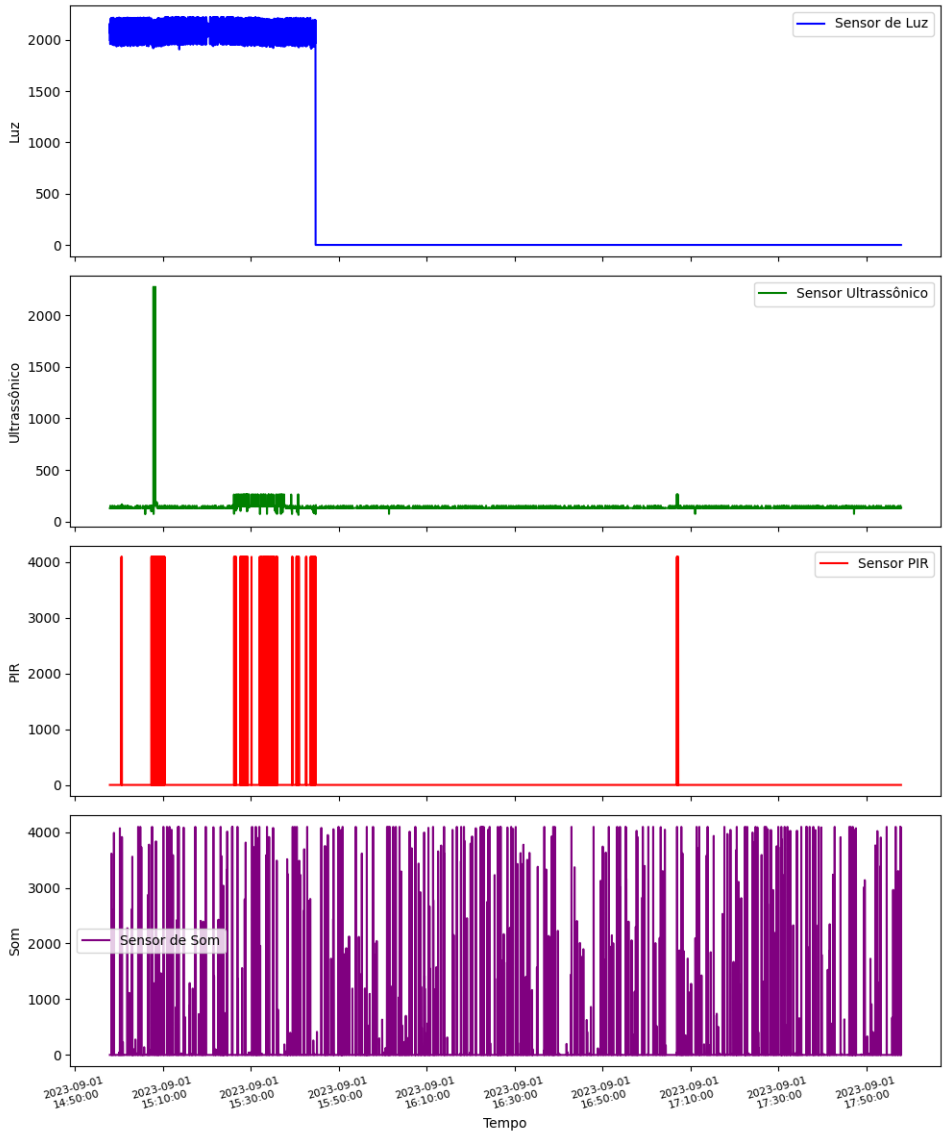

Figura 34 – Dados do dia 01/09/2023-4. Fonte: Própria

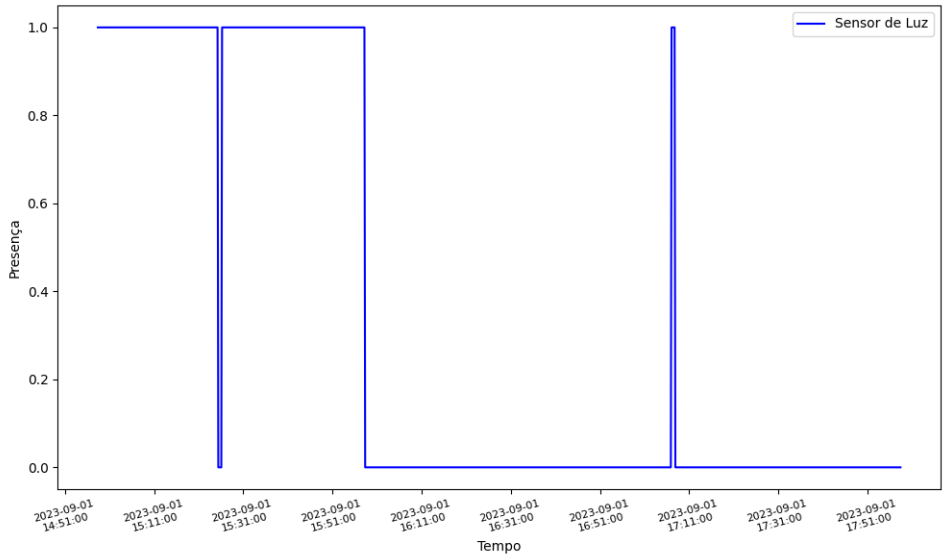

Figura 35 – Dados do Sensor Virtual do dia 01/09/2023-4. Fonte: Própria

4.5 Cenários de falhas nos sensores

A análise do desempenho de sensores é fundamental em diversas aplicações, e a garantia da sua confiabilidade é de grande importância. Nesse contexto, é crucial compreender como um sensor virtual proposto reage diante da ausência de informações de sensores físicos, ou seja, sua capacidade de tolerar falhas.

Com o objetivo de investigar esse aspecto, foram realizadas uma série de experimentos simulando cenários de falhas nos sensores. Essas falhas foram induzidas através da temporária desativação dos dados de sensores físicos específicos, permitindo-nos avaliar o comportamento do sensor virtual. Como resultado dos testes realizados, foram obtidos os dados dos sensores físicos e os dados do sensor virtual já desconsiderando um dos sensores físicos. A seguir, estão descritos esses testes de falhas realizados:

4.5.1 Falha no sensor ultrassônico

Os gráficos das Figuras 36 e 37 ilustram os resultados obtidos ao desabilitar o sensor ultrassônico. A Figura 36 apresenta o gráfico dos dados dos sensores físicos, enquanto a Figura 37 mostra os dados do sensor virtual, desconsiderando o sensor ultrassônico. Como observado, o sensor virtual ainda é capaz de manter tolerância a falhas, mantendo a detecção de presença relativamente estável.

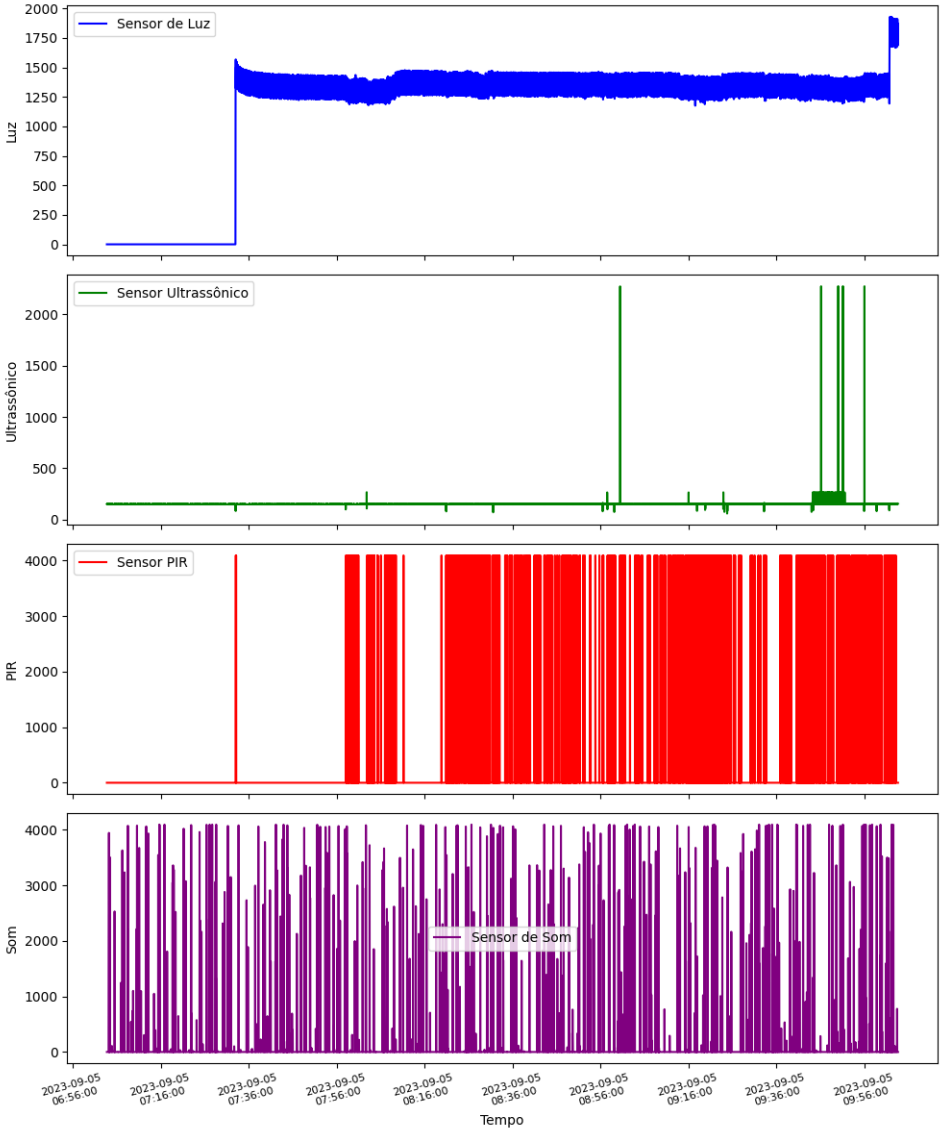

Figura 36 – Dados do dia 05/09/2023-1. Fonte: Própria

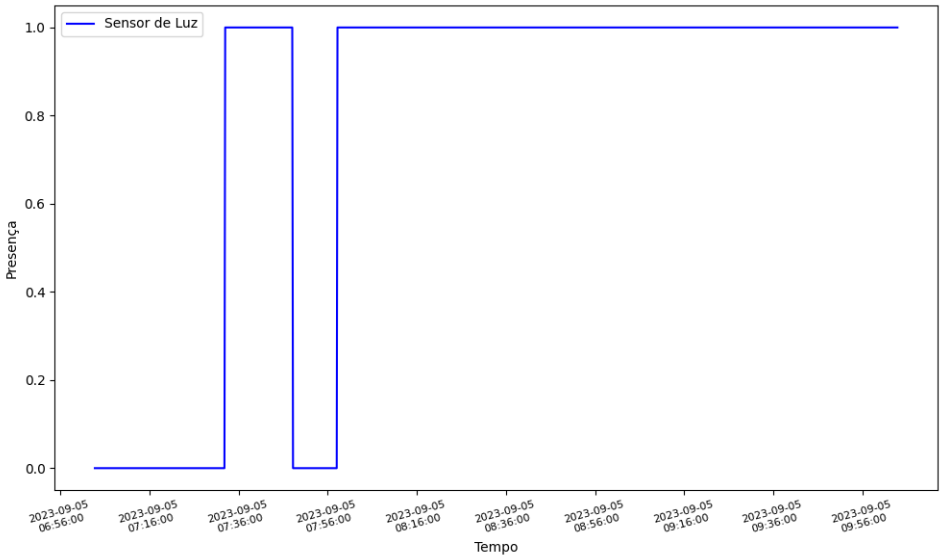

Figura 37 – Dados de teste de tolerância a falhas do Sensor Virtual do dia 05/09/2023-1. Fonte: Própria

4.5.2 Falha no sensor PIR

Ao desabilitar o sensor PIR, conforme mostrado nos gráficos das Figuras 38 e 39, observa-se que o sistema ainda é capaz de manter a detecção de presença, mantendo também o sensor tolerante a falhas.

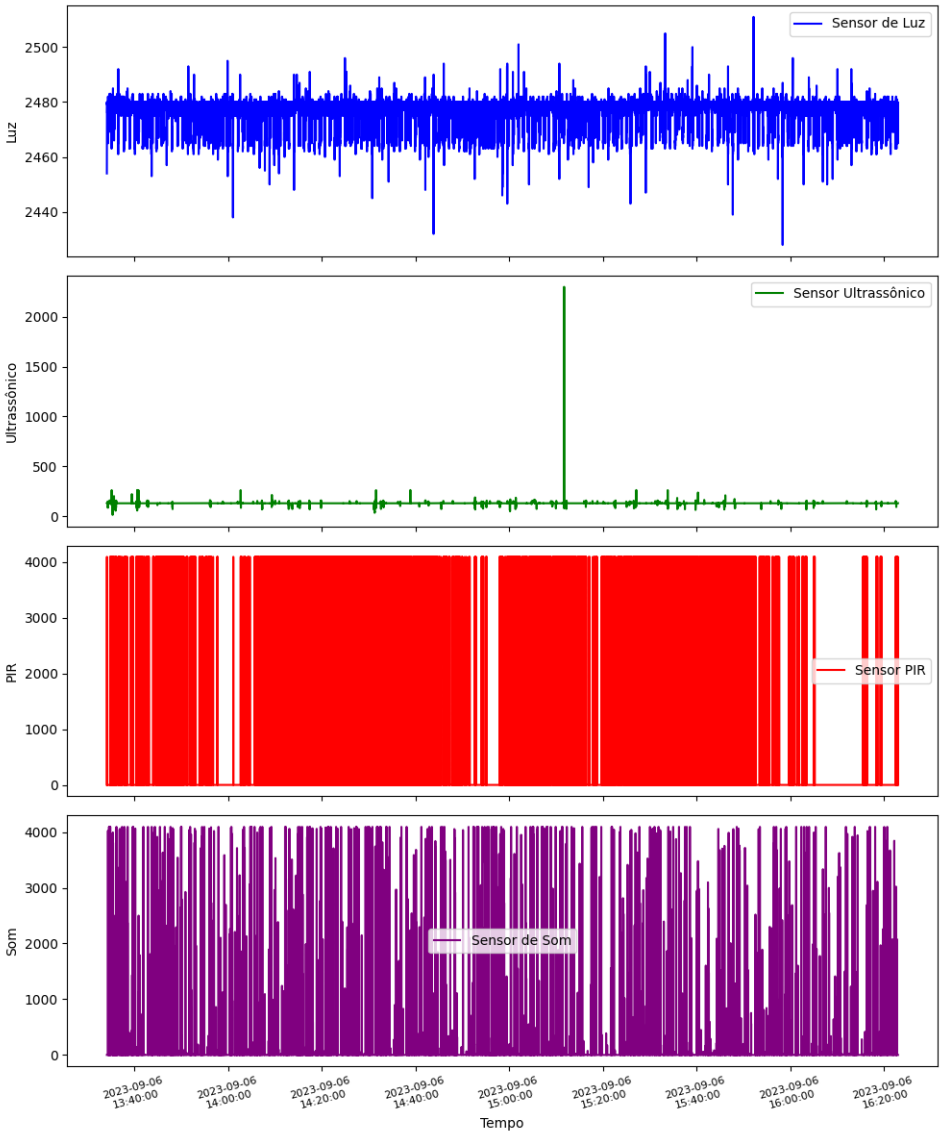

Figura 38 – Dados do dia 06/09/2023-2. Fonte: Própria

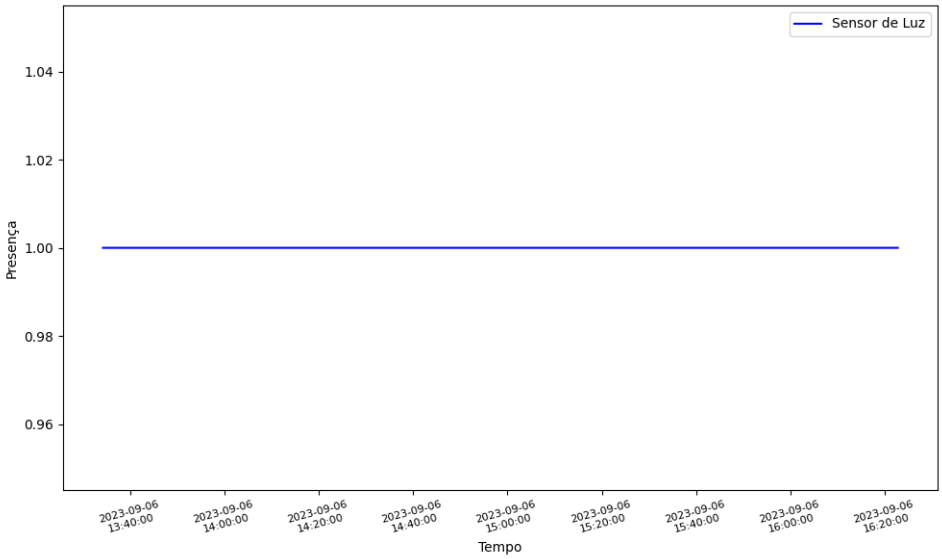

Figura 39 – Dados de teste de tolerância a falhas do Sensor Virtual do dia 06/09/2023-2. Fonte: Própria

4.5.3 Falha no sensor LDR

No entanto, ao desabilitar o sensor de luz, como demonstrado nos gráficos das Figuras 40 e 41, percebe-se uma variação significativa nos dados. Isso indica que a ausência do sensor de luz afeta a capacidade do sistema de manter a informação de presença no ambiente por um longo período de tempo.

Vale ressaltar que esses testes foram executados por um curto período de tempo, devido à restrição de tempo disponível. Infelizmente, não foi possível realizar testes por mais tempo. Em trabalhos futuros ou com mais tempo à disposição, seria desejável conduzir esses testes por um período mais prolongado, a fim de obter uma avaliação mais precisa da tolerância a falhas do sensor virtual.

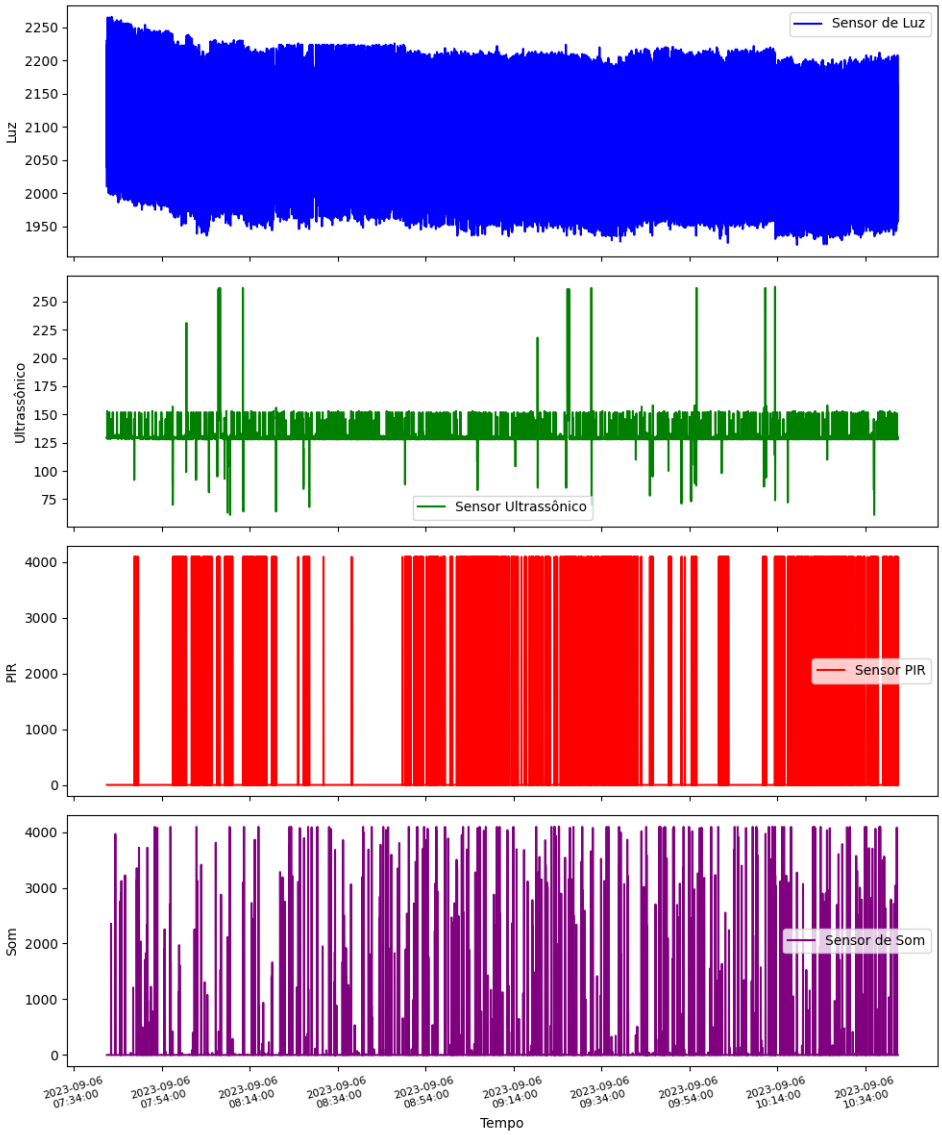

Figura 40 – Dados do dia 06/09/2023-1. Fonte: Própria

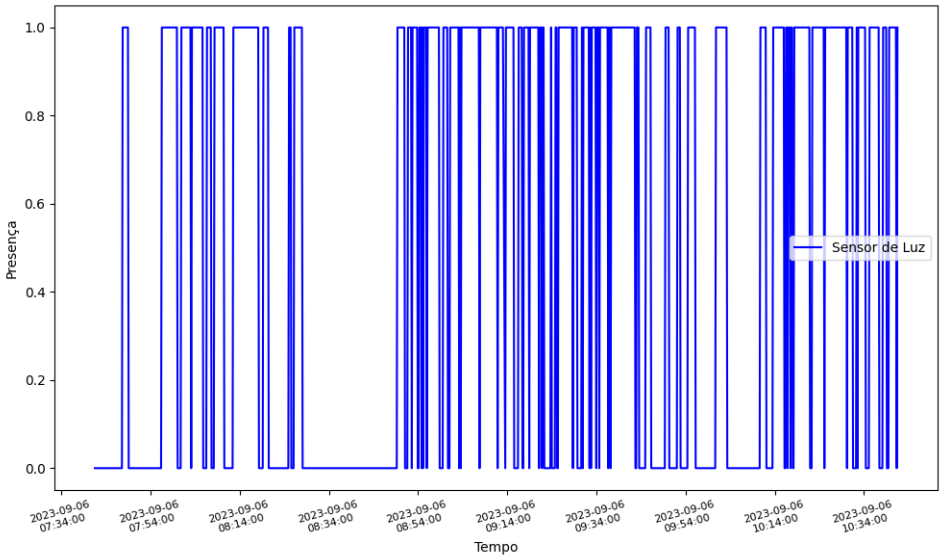

Figura 41 – Dados de teste de tolerância a falhas do Sensor Virtual do dia 06/09/2023-1. Fonte: Própria

5 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho teve como objetivo propor um sistema baseado em dispositivos IoT a fim de propor um sensor virtual capaz de monitorar a ocupação de salas e ambiente acadêmicos na UFPel buscando um melhor gerenciamento deste recurso. Com os resultados obtidos é possível concluir que o sistema e o sensor virtual proposto se mostraram eficientes e cumprem os objetivos do trabalho. A comunicação com o broker *MQTT* funcionou de maneira consistente, e os clientes conseguiram se conectar e fazer a comunicação com o servidor. Além disso o sensor virtual foi capaz de inferir as informações desejadas, demonstrando ser tolerante a falhas de relação a alguns sensores, com exceção do sensor de luz. Sem a contribuição do sensor de luz, o sensor virtual não é capaz de manter informações sobre a ocupação do ambiente por um longo período de tempo. No entanto, mesmo nessas condições, ele ainda é capaz de registrar momentos de presença e ausência no ambiente.

Como sugestão para trabalhos futuros, primeiramente, podem ser investigadas formas de melhorar o uso do sensor de som, que se mostrou ser muito ruidoso. Além disso, explorar alternativas nas configurações do sistema para torná-lo ainda mais tolerante a falhas pode ser uma direção promissora. Também vale a pena considerar a inclusão de outros tipos de sensores para melhorar a capacidade de detecção de presença no ambiente, e até diminuir a taxa de amostragem de alguns sensores, como o sensor de luminosidade, uma vez que os dados se mantêm quase sempre constantes.

Além disso, é importante observar que o sistema atualmente opera em um ambiente fechado e em uma rede local específica. No entanto, ao considerar a possibilidade de escalar o sistema para funcionar em toda a rede da UFPEL no campus, torna-se viável a implantação de vários módulos para monitorar diversas salas ou laboratórios. Isso ocorre porque o servidor com o broker *MQTT* é capaz de aceitar a conexões de vários clientes, permitindo a criação de vários tópicos para a troca de mensagens. Também pode-se considerar a criação de um cliente do tipo *subscriber* para coletar todas essas informações de vários ambientes e disponibilizá-las em um sistema, como uma página da web, assim permitindo uma visualização centralizada de todos os dados coletados, facilitando o monitoramento e a análise em tempo real.

Por fim, é importante considerar o uso mais amplo desta tecnologia, visando aumentar a eficiência energética em todo o campus, como identificar o funcionamento de equipamentos de iluminação, ventilação e refrigeração em salas desocupadas para reduzir gastos e promover mais o conceito de campus inteligentes e automação, possibilitando o desligamento automático desses equipamentos.

REFERÊNCIAS

AYDOS, M.; VURAL, Y.; TEKEREK, A. Assessing risks and threats with layered approach to Internet of Things security. **Measurement and Control**, [S.l.], v.52, p.002029401983799, 04 2019.

BENEDICT, S.; RUMAISE, P.; KAUR, J. IoT Blockchain Solution for Air Quality Monitoring in SmartCities. In: IEEE INTERNATIONAL CONFERENCE ON ADVANCED NETWORKS AND TELECOMMUNICATIONS SYSTEMS (ANTS), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.1–6.

CASADO-VARA, R.; VALE, Z.; PRIETO, J.; CORCHADO, J. M. Fault-tolerant temperature control algorithm for IoT networks in smart buildings. **Energies**, [S.l.], v.11, n.12, p.3430, 2018.

CHEN, F.; HUO, Y.; ZHU, J.; FAN, D. A review on the study on MQTT security challenge. In: IEEE INTERNATIONAL CONFERENCE ON SMART CLOUD (SMARTCLOUD), 2020., 2020. **Anais...** [S.l.: s.n.], 2020. p.128–133.

CRISTALDI, L. et al. Virtual Sensors: a Tool to Improve Reliability. In: IEEE INTERNATIONAL WORKSHOP ON METROLOGY FOR INDUSTRY 4.0 IOT, 2020., 2020. **Anais...** [S.l.: s.n.], 2020. p.142–145.

DATTA, S. K.; BONNET, C. Extending DataTweet IoT Architecture for Virtual IoT Devices. In: IEEE INTERNATIONAL CONFERENCE ON INTERNET OF THINGS (ITHINGS) AND IEEE GREEN COMPUTING AND COMMUNICATIONS (GREENCOM) AND IEEE CYBER, PHYSICAL AND SOCIAL COMPUTING (CPSCOM) AND IEEE SMART DATA (SMARTDATA), 2017., 2017. **Anais...** [S.l.: s.n.], 2017. p.689–694.

ESP32 Series - Datasheet. acessado em 25/08/2023, `https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf`.

FARIAS, C. M. de et al. Fusão de dados para Ambientes Inteligentes. **Sociedade Brasileira de Computação**, [S.l.], 2019.

FUNCIONAMENTO do sensor de presença PIR. acessado em 16/06/2023, https://www.daeletrica.com.br/blog/2018/10/24/funcionamento-do-sensor-de-presenca-pir/.

GONZALEZ, G. Soft sensors for processing plants. **Proceedings of the Second International Conference on Intelligent Processing and Manufacturing of Materials. IPMM'99 (Cat. No.99EX296)**, [S.l.], v.1, p.59–69 vol.1, 1999.

GROVE - Light Sensor. acessado em 16/06/2023, https://seeeddoc.github.io/Grove-Light_Sensor/.

GROVE - Sound Sensor. acessado em 16/06/2023, https://seeeddoc.github.io/Grove-Sound_Sensor/.

GUPTA, A.; MUKHERJEE, N. Implementation of virtual sensors for building a sensor-cloud environment. In: INTERNATIONAL CONFERENCE ON COMMUNICATION SYSTEMS AND NETWORKS (COMSNETS), 2016., 2016. **Anais...** [S.l.: s.n.], 2016. p.1–8.

GUPTA, A.; MUKHERJEE, N. Can the Challenges of IOT be Overcome by Virtual Sensors. In: IEEE INTERNATIONAL CONFERENCE ON INTERNET OF THINGS (ITHINGS) AND IEEE GREEN COMPUTING AND COMMUNICATIONS (GREENCOM) AND IEEE CYBER, PHYSICAL AND SOCIAL COMPUTING (CPSCOM) AND IEEE SMART DATA (SMARTDATA), 2017., 2017. **Anais...** [S.l.: s.n.], 2017. p.584–590.

GUPTA, A.; MUKHERJEE, N. A Cloudlet Platform With Virtual Sensors for Smart Edge Computing. **IEEE Internet of Things Journal**, [S.l.], v.6, n.5, p.8455–8462, 2019.

KUMAR, S.; CHANDRA, S. K.; SHUKLA, R. N.; PANIGRAHI, L. Industry 4.0 based Machine Learning Models for Anomalous Product Detection and Classification. In: OPJU INTERNATIONAL TECHNOLOGY CONFERENCE ON EMERGING TECHNOLOGIES FOR SUSTAINABLE DEVELOPMENT (OTCON), 2022., 2023. **Anais...** [S.l.: s.n.], 2023. p.1–6.

LIGHT, R. A. Mosquitto: server and client implementation of the MQTT protocol. **Journal of Open Source Software**, [S.l.], v.2, n.13, p.265, 2017.

LOTUFO, F.; GARCIA, C. Sensores Virtuais ou Soft Sensors: Uma introdução. , [S.l.], 01 2008.

MADAKAM, S. et al. Internet of Things (IoT): A literature review. **Journal of Computer and Communications**, [S.l.], v.3, n.05, p.164, 2015.

MARTIN, D.; KüHL, N.; SATZGER, G. Virtual Sensors. **Business Information Systems Engineering**, [S.l.], v.63, n.3, p.315–323, 2021.

MIN-ALLAH, N.; ALRASHED, S. Smart campus—A sketch. **Sustainable Cities and Society**, [S.l.], v.59, p.102231, 2020.

MUDASSAR, M.; ZHAI, Y.; LEJIAN, L. Adaptive Fault-Tolerant Strategy for Latency-Aware IoT Application Executing in Edge Computing Environment. **IEEE Internet of Things Journal**, [S.l.], v.9, n.15, p.13250–13262, 2022.

MóDULO Sensor Hc-sr501 de Presença e Movimento – PIR. acessado em 16/06/2023, https://www.institutodigital.com.br/produto/modulo-sensor-hc-sr501-de-presenca-e-movimento/.

O que é um sensor ultrassônico. acessado em 26/08/2023, https://www.mecanicaindustrial.com.br/598-o-que-e-um-sensor-ultrassonico/.

PASTÓRIO, A.; RODRIGUES, L.; CAMARGO, E. de. Uma revisao sistemática da literatura sobre tolerância a falhas em internet das coisas. **Anais Estendidos do X Simpósio Brasileiro de Engenharia de Sistemas Computacionais**, [S.l.], p.57–64, 2020.

PATEL, K. K.; PATEL, S. M.; SCHOLAR, P. Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges. **International journal of engineering science and computing**, [S.l.], v.6, n.5, 2016.

PENIAK, P.; BUBENíKOVá, E.; KANáLIKOVá, A. The Redundant Virtual Sensors via Edge Computing. In: INTERNATIONAL CONFERENCE ON APPLIED ELECTRONICS (AE), 2021., 2021. **Anais...** [S.l.: s.n.], 2021. p.1–5.

PING, J.; LIU, Y.; WENG, D. Comparison in Depth Perception between Virtual Reality and Augmented Reality Systems. In: 2021 , 2019. **Anais...** [S.l.: s.n.], 2019. p.1124–1125.

RAVEENDRANATHAN, N. et al. From Modeling to Implementation of Virtual Sensors in Body Sensor Networks. **IEEE Sensors Journal**, [S.l.], v.12, n.3, p.583–593, 2012.

SANTOS, B. P.; ALBERTO, A.; LIMA, T. D. F. M.; CHARRUA-SANTOS, F. M. B. Indústria 4.0: desafios e oportunidades. **Revista Produção e Desenvolvimento**, [S.l.], v.4, n.1, p.111–124, 2018.

SCHWAB, K. **A quarta revolução industrial**. [S.l.]: Edipro, 2019.

SHI, W. et al. Edge Computing: Vision and Challenges. **IEEE Internet of Things Journal**, [S.l.], v.3, n.5, p.637–646, 2016.

SITTÓN-CANDANEDO, I. et al. Edge computing, iot and social computing in smart energy scenarios. **Sensors**, [S.l.], v.19, n.15, p.3353, 2019.

SONI, D.; MAKWANA, A. A survey on mqtt: a protocol of internet of things (iot). In: INTERNATIONAL CONFERENCE ON TELECOMMUNICATION, POWER ANALYSIS AND COMPUTING TECHNIQUES (ICTPACT-2017), 2017. **Anais...** [S.l.: s.n.], 2017. v.20, p.173–177.

Wikipedia contributors. **Photoresistor — Wikipedia, The Free Encyclopedia**. [Online; accessed 16-June-2023], https://en.wikipedia.org/w/index.php?title=Photoresistor&oldid=1148374096.

YIN, C. et al. A literature survey on smart cities. **Sci. China Inf. Sci.**, [S.l.], v.58, n.10, p.1–18, 2015.

ZHOU, S. et al. Supporting service adaptation in fault tolerant internet of things. In: IEEE 8TH INTERNATIONAL CONFERENCE ON SERVICE-ORIENTED COMPUTING AND APPLICATIONS (SOCA), 2015., 2015. **Anais...** [S.l.: s.n.], 2015. p.65–72.