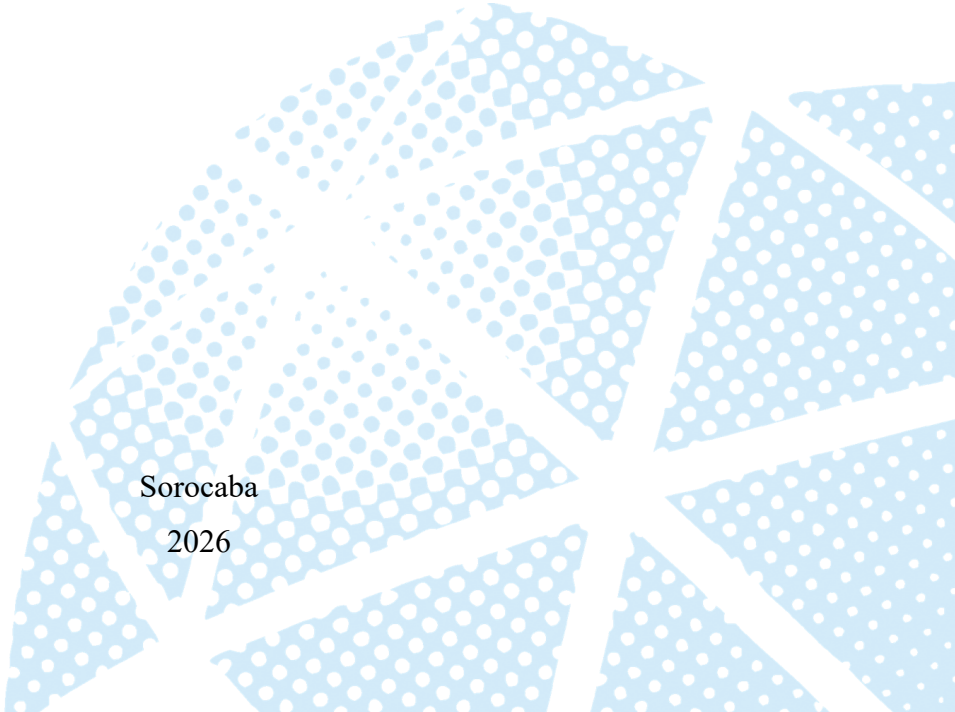
UNIVERSIDADE ESTADUAL PAULISTA - UNESP

Instituto de Ciência e Tecnologia - Campus de Sorocaba

Matheus River de Souza

Desenvolvimento de infraestrutura modular para monitoramento de processos e diagnóstico de dispositivos na indústria 4.0



Sorocaba

2026

Matheus River de Souza

Desenvolvimento de infraestrutura modular para monitoramento de processos e diagnóstico de dispositivos na indústria 4.0

Trabalho de Conclusão de Curso apresentado à Universidade Estadual Paulista (UNESP), Instituto de Ciência e Tecnologia, Sorocaba, como parte dos requisitos para obtenção do grau de Bacharel em Engenharia de Controle e Automação.

Orientador: Prof. Dr. Ricardo Pasquati Pontarolli

Coorientador: Prof. Dr. Eduardo Paciência Godoy

Sorocaba

2026

S729d

Souza, Matheus River de

Desenvolvimento de infraestrutura modular para monitoramento de processos e diagnóstico de dispositivos na indústria 4.0 / Matheus River de Souza. -- Sorocaba, 2026

55 p. : tabs., fotos

Trabalho de conclusão de curso (Bacharelado - Engenharia de Controle e Automação) - Universidade Estadual Paulista (UNESP), Instituto de Ciência e Tecnologia, Sorocaba

Orientador: Ricardo Pasquati Pontarolli

Coorientador: Eduardo Paciência Godoy

1. Banco de dados temporais. 2. Análise de séries temporais. 3. Dashboards (Sistemas de informação gerencial). I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Dados fornecidos pelo autor(a).

Matheus River de Souza

Desenvolvimento de infraestrutura modular para monitoramento de processos e diagnóstico de dispositivos na indústria 4.0

Trabalho de Conclusão de Curso apresentada à Universidade Estadual Paulista (UNESP), Instituto de Ciência e Tecnologia - Câmpus de Sorocaba, para obtenção do título de Bacharel(a) em Engenharia de controle e automação.

Data da defesa: 22/07/2026

Banca Examinadora:

_____

Prof. Dr. Ricardo Pasquati Pontarolli

IFSP - Instituto Federal de Educação, Ciência e Tecnologia de São Paulo - Campus Boituva

_____

Prof. Dr. Eduardo Verri Liberado

UNESP - Instituto de Ciência e Tecnologia - Campus de Sorocaba

_____

Prof. Dr. Everson Martins

UNESP - Instituto de Ciência e Tecnologia - Campus de Sorocaba

AGRADECIMENTOS

Agradeço aos meus pais, Lilian Lima e Solivã Souza, por sempre me incentivarem na escolha deste curso e por me apoiarem incondicionalmente em cada etapa dessa jornada. À minha namorada, Beatriz Pedroso, por estar ao meu lado nos momentos de maior dificuldade ao longo do curso e por me ajudar a seguir em frente. Aos meus amigos, João Castro e Alan Tonon, pelo apoio constante e pelas madrugadas de estudo. Ao Prof. Dr. Eduardo P. Godoy, pela oportunidade de desenvolver meu primeiro projeto de iniciação científica, e ao meu orientador, Prof. Dr. Ricardo P. Pontarolli, pelo apoio constante no desenvolvimento deste projeto.

O presente trabalho foi realizado com apoio da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), Brasil. Processo nº 2024/21954-7.

“As invenções são, sobretudo, o resultado de um trabalho teimoso.”

(Alberto Santos Dumont, 1918).

RESUMO

A crescente adoção de sistemas distribuídos na Indústria 4.0 tem ampliado a necessidade de soluções capazes de armazenar, organizar e disponibilizar grandes volumes de dados gerados continuamente por sensores, atuadores e dispositivos computacionais. Essas informações são fundamentais para a análise de desempenho, identificação de falhas, rastreabilidade e suporte à tomada de decisão. Em arquiteturas distribuídas, como a *Plug-and-Produce* (PnPr), o acompanhamento contínuo desses dados são essenciais para o monitoramento do comportamento de plantas industriais. Em trabalhos anteriores, foi desenvolvida uma arquitetura PnPr orientada a microsserviços para uma planta piloto de Indústria 4.0. Entretanto, essa arquitetura não contemplava mecanismos para o armazenamento histórico e o monitoramento integrado das variáveis de processo da planta e das métricas de hardware. Para suprir essa limitação, este trabalho propõe o desenvolvimento de uma infraestrutura modular para monitoramento de processos e diagnóstico de dispositivos, integrada à arquitetura PnPr. O objetivo foi desenvolver e integrar essa infraestrutura, possibilitando a aquisição, o armazenamento e a visualização histórica dispositivos que compõem a planta. Para isso, foi desenvolvida uma infraestrutura baseada nas ferramentas Node-RED, Telegraf, InfluxDB e Grafana, responsável pela aquisição, armazenamento e visualização histórica dessas variáveis de processo e das métricas de hardware da planta piloto. Os resultados demonstraram que a infraestrutura foi integrada com sucesso ao PnPr, permitindo o monitoramento unificado das variáveis de processo e das métricas de hardware por meio de uma única plataforma de supervisão. Além disso, a solução mostrou-se escalável, permitindo a incorporação de novos dispositivos com reduzido esforço de configuração.

Palavras-chave: InfluxDB; séries temporais; Grafana; Historiador de dados; Plug-and-Produce; microsserviços.

ABSTRACT

The increasing adoption of distributed systems in Industry 4.0 has intensified the need for solutions capable of storing, organizing, and providing access to the large volumes of data continuously generated by sensors, actuators, and computing devices. This information is essential for performance analysis, fault identification, traceability, and decision-making support. In distributed architectures, such as Plug-and-Produce (PnPr), the continuous monitoring of this data is essential for supervising the behavior of industrial plants. In previous studies, a microservice-oriented PnPr architecture was developed for an Industry 4.0 pilot plant. However, this architecture did not include mechanisms for the historical storage and integrated monitoring of process variables and hardware metrics. To address this limitation, this work proposes the development of a modular infrastructure for process monitoring and device diagnostics, integrated into the PnPr architecture. The objective was to develop and integrate this infrastructure, enabling the acquisition, storage, and historical visualization of data from the devices composing the pilot plant. To achieve this, an infrastructure based on Node-RED, Telegraf, InfluxDB, and Grafana was developed to provide the acquisition, storage, and historical visualization of process variables and hardware metrics. The results demonstrated that the proposed infrastructure was successfully integrated into the PnPr architecture, enabling unified monitoring of process variables and hardware metrics through a single supervision platform. Furthermore, the solution proved to be scalable, allowing the incorporation of additional devices with minimal configuration effort.

Keywords: InfluxDB; time-series database; Grafana; Data Historian; Plug-and-Produce; microservices.

LISTA DE ILUSTRAÇÕES

Figura 1 – Infraestrutura de monitoramento 19
Figura 2 – Vista frontal e P&ID da planta piloto 4.0 21
Figura 3 – Arquitetura Orientada a Microsserviços do Moleculer 22
Figura 4 – Placa Raspberry Pi 3B+lado esquerdoe MegaIOIndustrialao lado direito 24
Figura 5 – Raspberry Pi imager 27
Figura 6 – Containers instalados 28
Figura 7 – Interface de gerenciamento Portainer.io 29
Figura 8 – Diagrama de sequência Monitoramento das Variáveis de Processo 30
Figura 9 – Diagrama de sequência Monitoramento das Métricas de Hardware 31
Figura 10 – Estrutura banco de dados 33
Figura 11 – Processamento da resposta Microsserviço 34
Figura 12 – Flow de orquestração Node-RED 35
Figura 13 – Plugin de conexão Telegraf com o Node-RED 36
Figura 14 – Escrita das métricas de hardware no InfluxDB 37
Figura 15 – Conexão Grafana e InfluxDB 38
Figura 16 – Páginas Dashboard 39
Figura 17 – Menu Dashboard 40
Figura 18 – Dashooard Historian Home 41
Figura 19 – Dashboard hardware monitoring 42
Figura 20 – Seleção do dispositivo monitorado 43
Figura 21 – Variável Grafana host 43
Figura 22 – Configuração de time range 44
Figura 23 – Requisição de uso de CPU 45
Figura 24 – Dashboard control monitoring 47
Figura 25 – Variável Grafana control mesh 48
Figura 26 – Variável Grafana sensor 48
Figura 27 – Display planta piloto Produce 49
Figura 28 – Gráfico de controle obtido no Grafana 52

LISTA DE TABELAS

Tabela 1 – Variáveis de processo no InfluxDB 34
Tabela 2 – Métricas de hardware utilizadas 36
Tabela 3 – Tempo médio de execução das consultas dos painéis do Control Monitoring 50
Tabela 4 – Tempo médio de execução das consultas dos painéis do Hardware Monitoring 51

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
CPU	*Central Processing Unit*
I4.0	Indústria 4.0
IAE	*Integral of Absolute Error*
IIoT	*Industrial Internet of Things*
IoT	*Internet of Things*
IP	*Internet Protocol*
ISE	*Integral of Squared Error*
MV	*Manipulated Variable*
NoSQL	*Not Only SQL*
PnP	*Plug-and-Play*
PnPr	*Plug-and-Produce*
P&ID	*Piping and Instrumentation Diagram*
PnP	*Plug-and-Play*
PV	*Process Variable*
RAM	*Random Access Memory*
SP	*Setpoint*
SQL	*Structured Query Language*
SSD	*Solid State Drive*
TSDB	*Time Series Database*

SUMÁRIO

1 INTRODUÇÃO 13
1.1 Objetivos 14
1.2 Estrutura do Trabalho 14
2 REVISÃO CONCEITUAL 16
2.1 Banco de dados para IIOT 16
2.2 Historiador de processos 17
3 METODOLOGIA 19
3.1 Proposta de trabalho 19
3.2 Materiais 20
3.2.1 Planta Piloto 4.0 de Microsserviços 20
3.2.2 Framework Moleculer e Microsserviços disponíveis 22
3.2.3 Hardware 23
3.2.4 Node-RED 24
3.2.5 InfluxDB 25
3.2.6 Telegraf 25
3.2.7 Grafana 26
3.3 Métodos 26
4 DESENVOLVIMENTO 30
4.1 Fluxo de monitoramento das variáveis de processo 30
4.2 Fluxo de monitoramento das métricas de hardware 31
4.3 Implementação do Banco de Dados 32
4.4 Implementação da Orquestração 33
4.5 Implementação do Monitoramento de Hardware 35
4.6 Desenvolvimento dos Dashboards 37
4.6.1 Dashboard Home 40
4.6.2 Dashboard Hardware Monitoring 41
4.6.3 Dashboard Control Monitoring 45
4.7 Integração com a arquitetura PnPr 49
5 RESULTADOS E DISCUSSÕES 50
6 CONCLUSÃO 53
REFERÊNCIAS 54

1 INTRODUÇÃO

A indústria 4.0 (I4.0) tem promovido transformações significativas na forma como os sistemas de automação e controle são concebidos, sobretudo no que diz respeito à integração de sensores, atuadores e controladores em ambientes industriais distribuídos (SENNA et al., 2022). Tecnologias emergentes como a internet das coisas industrial (IIoT), a computação de borda e arquiteturas baseadas em microsserviços vêm estabelecendo uma nova infraestrutura para o chão de fábrica, com ênfase em conectividade, interoperabilidade e integração vertical dos processos produtivos (CARLSSON et al., 2018).

A crescente utilização dessas tecnologias resulta na geração contínua de grandes volumes de dados provenientes de sensores, atuadores e dispositivos computacionais distribuídos. O armazenamento e a análise dessas informações são fundamentais para o acompanhamento do comportamento dos processos industriais, identificação de falhas, rastreabilidade e suporte à tomada de decisão (SILVA, 2023). Nesse contexto, os historiadores de processo desempenham papel essencial ao registrar continuamente as variáveis operacionais de plantas industriais, permitindo sua consulta e análise ao longo do tempo (DOMINGUES, 2021). Para atender às elevadas taxas de aquisição características dessas aplicações, bancos de dados orientados a séries temporais destacam-se como uma solução adequada para o armazenamento e processamento eficiente dessas informações (PIRES, 2025).

Entre as arquiteturas distribuídas utilizadas na I4.0, destacam-se as soluções baseadas em microsserviços, que oferecem uma abordagem modular para o desenvolvimento de aplicações industriais. Essa arquitetura proporciona vantagens como escalabilidade, resiliência, modularidade e independência no desenvolvimento e implantação dos serviços (PONTAROLLI et al., 2023). Quando combinada à computação de borda, possibilita reduzir a latência das comunicações e distribuir o processamento entre diferentes dispositivos da infraestrutura.

Como uma aplicação dessa abordagem, destaca-se a arquitetura Plug-and-Produce (PnPr), desenvolvida para permitir a integração flexível e autônoma de dispositivos e serviços industriais. Diferentemente do conceito de Plug-and-Play (PnP), consolidado no domínio da computação, o PnPr ainda se encontra em fase de consolidação, carecendo de normas e procedimentos que orientem sua implementação prática em ambientes industriais (KHABBAZI et al., 2024; PONTAROLLI et al., 2024).

Apesar dos avanços proporcionados pela arquitetura Plug-and-Produce, a solução baseada em microsserviços desenvolvida por Pontarolli (2025) não contemplava mecanismos

para o monitoramento contínuo e o armazenamento histórico das variáveis de processo e das métricas de hardware dos dispositivos que a compõem. A ausência dessa camada dificulta o acompanhamento do comportamento da planta e do desempenho computacional da arquitetura ao longo do tempo, limitando a identificação de falhas e a análise operacional.

Diante dessa limitação, este trabalho propõe a integração de uma infraestrutura modular de monitoramento à arquitetura PnPr previamente desenvolvida, utilizando as ferramentas Telegraf, InfluxDB, Grafana e Node-RED para realizar a aquisição, o armazenamento e a visualização histórica das variáveis de processo e das métricas de hardware de uma planta piloto desenvolvida para simular processos da I4.0. Com essa integração, espera-se ampliar a capacidade de supervisão da arquitetura, fornecendo subsídios para análise de desempenho, diagnóstico de dispositivos e apoio à tomada de decisão em aplicações alinhadas aos princípios da Indústria 4.0.

1.1 Objetivos

O objetivo deste trabalho é desenvolver e integrar uma camada de monitoramento distribuído à arquitetura PnPr baseada em microsserviços, previamente desenvolvida para aplicações de automação e controle na I4.0. A solução proposta visa possibilitar a aquisição, armazenamento e visualização histórica de variáveis de processo e métricas de hardware dos dispositivos que compõem a infraestrutura da planta piloto.

Com a implementação dessa infraestrutura, busca-se ampliar a capacidade de supervisão da arquitetura, permitindo o acompanhamento contínuo do desempenho computacional dos dispositivos e do comportamento operacional da planta, contribuindo para a identificação de falhas, análise de desempenho e suporte à tomada de decisão.

1.2 Estrutura do Trabalho

Este trabalho está organizado em seis capítulos, além dos elementos pré-textuais e do resumo. O Capítulo 1 apresenta a introdução ao tema, contextualizando o problema abordado, a motivação para o desenvolvimento da pesquisa e os objetivos propostos. O Capítulo 2 contém a revisão conceitual, abordando os fundamentos necessários para a compreensão do trabalho.

No Capítulo 3 é apresentada a metodologia adotada, incluindo a descrição detalhada da proposta de trabalho, o fluxo de funcionamento da infraestrutura e os materiais e métodos

empregados durante a implementação. O Capítulo 4 descreve o desenvolvimento da solução proposta, detalhando as etapas de configuração, integração e implementação dos componentes que compõem a camada de monitoramento.

No Capítulo 5 são apresentados os resultados obtidos, bem como as discussões relacionadas ao desempenho da solução desenvolvida e às possibilidades de trabalhos futuros. Por fim, o Capítulo 6 apresenta as conclusões do trabalho, destacando os objetivos alcançados, as contribuições obtidas e as perspectivas para futuras evoluções da arquitetura proposta.

2 REVISÃO CONCEITUAL

Esta seção apresenta uma revisão dos principais conceitos que fundamentam o desenvolvimento deste trabalho. Para isso, o capítulo está dividido em duas subseções. A primeira aborda os bancos de dados voltados para aplicações de IIoT, com ênfase nos bancos de dados de séries temporais e suas características. A segunda apresenta os conceitos de historiadores de processos industriais e discute trabalhos relacionados, destacando suas principais contribuições e limitações.

2.1 Banco de dados para IIOT

Uma série temporal é caracterizada como uma sequência de observações coletadas e organizadas cronologicamente, na qual a ordem temporal dos dados é essencial para a análise das relações entre eventos passados e presentes (GUEDES, 2025). Com o avanço da I4.0 e a expansão da IIoT, um número cada vez maior de sensores passou a estar conectado à internet, resultando em um crescimento significativo do volume de dados temporais gerados. Esse cenário impulsionou o desenvolvimento dos bancos de dados orientados a séries temporais (*Time Series Databases* – TSDB), projetados especificamente para oferecer alta eficiência no armazenamento, processamento e consulta de dados indexados por tempo (LIMA; LICHTNOW, 2024).

Diferentemente dos bancos de dados relacionais (SQL) e dos bancos não relacionais (NoSQL) de propósito geral, os TSDB são otimizados para aplicações que realizam escrita contínua de dados, consultas baseadas em intervalos de tempo e armazenamento de grandes volumes de séries temporais, características comuns em sistemas de IIoT (DOMINGUES, 2021).

Em PIRES (2025), foi realizada uma comparação entre bancos de dados SQL, NoSQL e de TSDB aplicados a cenários de I4.0. O estudo utilizou o InfluxDB como representante dos TSDB e demonstrou que esse banco apresentou menor latência média, maior taxa de ingestão de dados e melhor escalabilidade quando comparado aos SQL e NoSQL. Esses resultados indicam que os TSDB são mais adequados para aplicações de IIoT, nas quais há geração contínua e armazenamento de grandes volumes de dados indexados temporalmente, exigindo elevado desempenho nas operações de escrita e consulta.

2.2 Historiador de processos

Os historiadores de processos industriais passaram a ser amplamente utilizados a partir da década de 1980, tornando-se ferramentas essenciais para o armazenamento e gerenciamento de dados históricos em plantas industriais. Atualmente, são empregados em diversos setores da indústria para registrar variáveis de processo, permitindo a análise do desempenho operacional, a identificação de falhas, o suporte à manutenção preditiva e a otimização dos processos produtivos. Além disso, uma nova geração de historiadores passou a ser incorporada diretamente em controladores e dispositivos de campo, possibilitando a aquisição imediata dos dados e a atribuição precisa de marcações temporais (*timestamps*), aumentando a confiabilidade e a precisão dos registros históricos (LYNDON, 2017).

Os historiadores modernos são frequentemente implementados com o auxílio de bancos de dados orientados a séries temporais, capazes de armazenar grandes volumes de informações indexadas temporalmente. Associados a plataformas de visualização, esses sistemas permitem acompanhar tanto o comportamento instantâneo quanto a evolução histórica das variáveis monitoradas (INFLUXDATA, 2025).

A aplicação desse conceito pode ser observada em diferentes trabalhos da literatura. Domingues (2021), por exemplo, desenvolveu um historiador de processos para uma planta piloto de manufatura utilizando o banco de dados InfluxDB para o armazenamento das séries temporais e a plataforma Grafana para a visualização e análise histórica das variáveis de processo. A solução possibilitou o acompanhamento em tempo real e a consulta aos dados registrados, fornecendo uma infraestrutura para supervisão e análise operacional da planta.

De forma semelhante, Santana (2022) desenvolveu um sistema de monitoramento para uma planta piloto baseado em realidade aumentada, no qual foi construída uma representação tridimensional da planta acessível por meio de um aplicativo para dispositivos Android. A solução permitiu a visualização das informações relacionadas ao processo de controle em um ambiente interativo, ampliando as possibilidades de supervisão e interação com a planta industrial.

Embora as soluções propostas por Domingues (2021) e Santana (2022) tenham contribuído para o monitoramento e a visualização das variáveis de processo, ambas concentraram seu escopo exclusivamente nesse tipo de informação. Dessa forma, aspectos relacionados ao monitoramento da infraestrutura computacional responsável pela execução da arquitetura distribuída, como utilização de CPU, memória, armazenamento e temperatura dos dispositivos, não foram contemplados. Essa limitação evidencia a necessidade de soluções

capazes de integrar, em uma mesma infraestrutura de monitoramento, tanto as variáveis de processo quanto as métricas de hardware dos dispositivos, proporcionando uma visão mais abrangente do comportamento da planta e da infraestrutura que a suporta.

3 METODOLOGIA

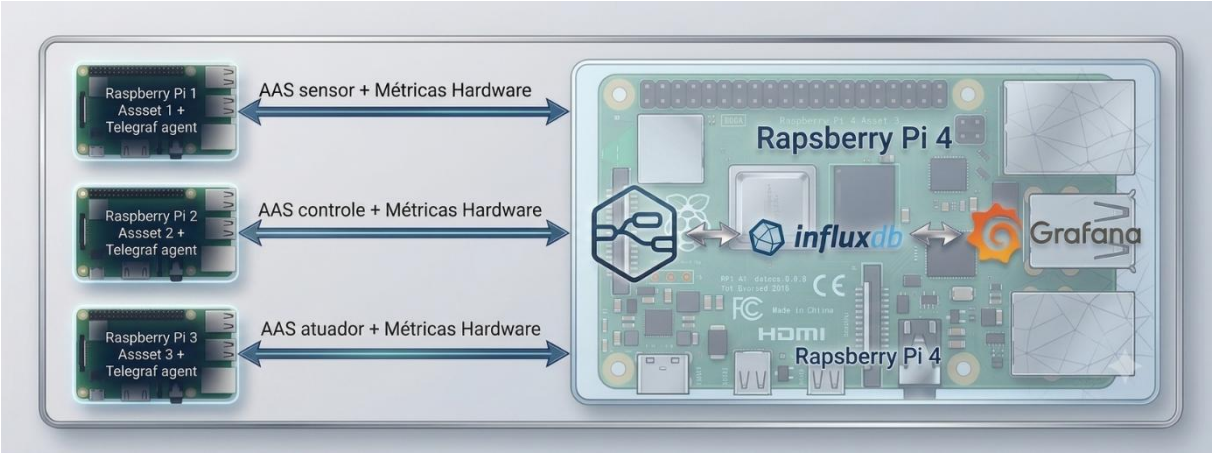
Esta seção apresenta a metodologia adotada para o desenvolvimento deste trabalho. Inicialmente, é descrita a proposta de trabalho, detalhando a infraestrutura de monitoramento desenvolvida. Em seguida, são apresentados os materiais e métodos utilizados durante o projeto.

3.1 Proposta de trabalho

A proposta deste trabalho consiste na expansão da arquitetura *Plug-and-Produce* previamente desenvolvida, por meio da integração de uma camada de monitoramento distribuído e visualização de dados em tempo real, conforme ilustrado na Figura 1. Para isso, serão utilizadas as placas Raspberry Pi já incorporadas à arquitetura, nas quais serão executadas as aplicações responsáveis pela aquisição das variáveis de processo da planta industrial e de métricas de desempenho dos dispositivos que compõem a infraestrutura.

Os dados coletados serão organizados em séries temporais, contendo marcações temporais e metadados que identificam sua origem e contexto. Em seguida, essas informações serão armazenadas em um banco de dados, tornando-se disponíveis para consulta e visualização por meio de painéis de controle (*dashboards*), que permitirão o monitoramento em tempo real e o historiamento das variáveis de processo e das métricas da arquitetura.

Figura 1 – Infraestrutura de monitoramento



Fonte: Elaborado pelo autor

A infraestrutura proposta é dividida em dois eixos principais de monitoramento. O primeiro é destinado à aquisição e ao armazenamento das variáveis de processo provenientes dos ativos (*assets*) da planta industrial, como sensores, atuadores e controladores. A coleta dessas informações será realizada por um orquestrador, responsável por acessar os microsserviços disponibilizados pelos *assets*, obter os dados de processo e registrá-los no banco de dados de séries temporais.

O segundo eixo é responsável pelo monitoramento dos recursos computacionais dos dispositivos que compõem a arquitetura distribuída, permitindo acompanhar indicadores como utilização de CPU, memória RAM, armazenamento e demais métricas de desempenho. A aquisição dessas informações será realizada por agentes coletores de métricas instalados nas placas Raspberry Pi, que enviarão periodicamente os dados para o banco de dados, possibilitando o acompanhamento contínuo do estado operacional da infraestrutura.

Com a integração desses dois eixos de monitoramento, espera-se centralizar as informações em um único banco de dados, possibilitando o armazenamento, a consulta e a visualização integrada dos dados gerados pela arquitetura PnPr. Dessa forma, será possível acompanhar, de maneira integrada, o comportamento das variáveis de processo e o desempenho computacional dos dispositivos distribuídos, fornecendo suporte ao monitoramento contínuo da infraestrutura, à análise histórica dos dados e à avaliação do desempenho da arquitetura.

3.2 Materiais

Nesta subseção são apresentadas as ferramentas e tecnologias que compõem a solução desenvolvida, bem como suas respectivas funções e aplicações no contexto da arquitetura proposta.

3.2.1 Planta Piloto 4.0 de Microsserviços

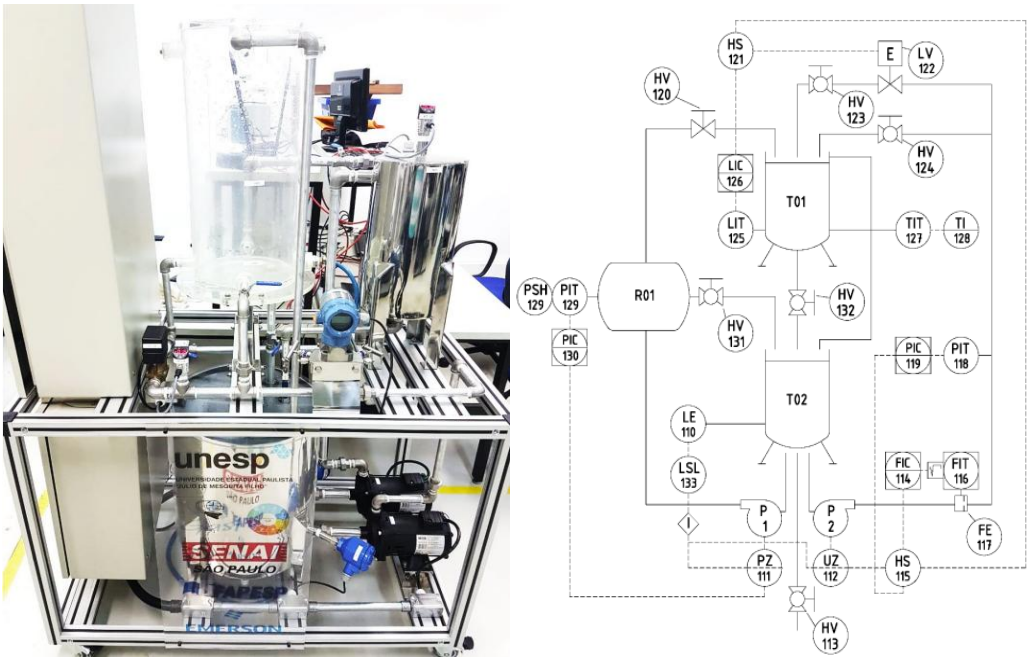
A planta piloto 4.0 utilizada neste trabalho constitui um ambiente experimental voltado à automação e controle de processos industriais alinhados aos conceitos da Indústria 4.0, sendo caracterizada pela utilização de uma arquitetura baseada em microsserviços (PONTAROLLI et al., 2020). A estrutura da planta permite a implementação e validação de aplicações distribuídas em um cenário próximo ao encontrado em ambientes industriais reais.

Fisicamente, a planta é composta por um reservatório principal em aço inoxidável localizado na parte inferior da estrutura, responsável pelo armazenamento de água. O fluido

pode ser transportado para a parte superior da planta por meio de duas rotas distintas: uma direcionada ao tanque em acrílico e outra ao reservatório pressurizado em inox, ambas acionadas por bombas independentes.

A infraestrutura elétrica da planta é dividida em dois painéis. O primeiro é destinado aos circuitos de potência e comando, enquanto o segundo concentra as interfaces de entrada e saída (IO), módulos programáveis e dispositivos de supervisão. A Figura 2 apresenta a vista frontal da planta piloto, bem como seu respectivo Diagrama de Processo e Instrumentação (P&ID).

Figura 2 – Vista frontal e P&ID da planta piloto 4.0



Fonte: (PONTAROLLI et al., 2020)

As principais variáveis de processo disponibilizadas pela planta são nível, vazão, pressão e temperatura, medidas respectivamente pelos instrumentos LIT125, FIT116, PIT118, PIT119 e TIT127. Como elementos de atuação, o sistema possui duas bombas acionadas por inversores de frequência e uma válvula proporcional, permitindo a implementação de estratégias de controle em diferentes condições operacionais.

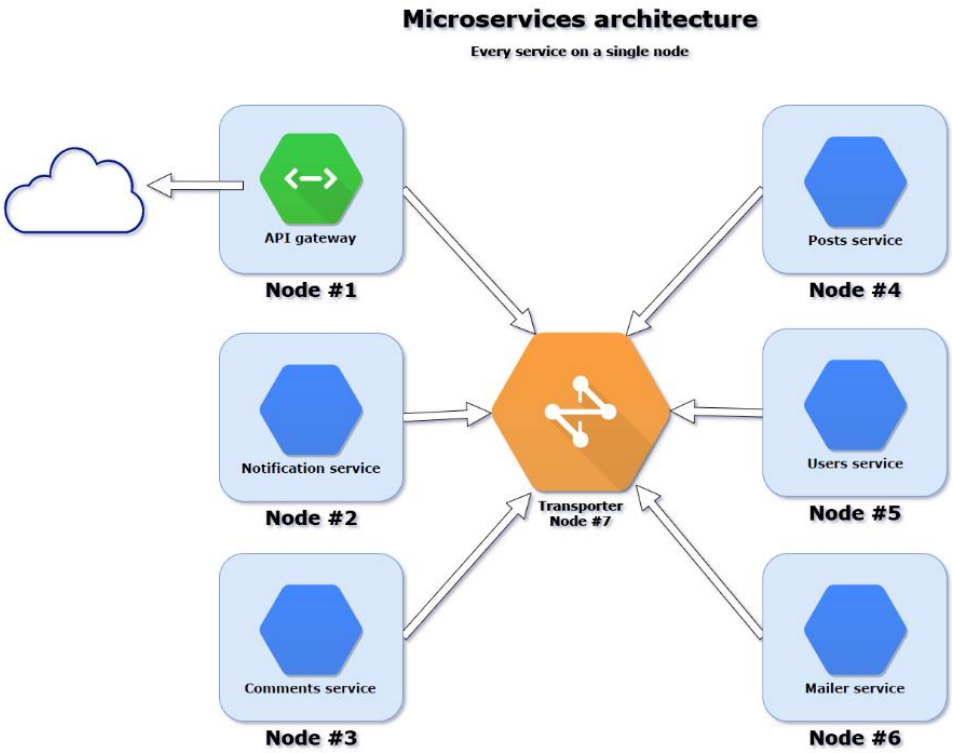
Além de sua função didática e experimental, a planta piloto fornece uma infraestrutura adequada para integração de aplicações distribuídas, aquisição de dados e monitoramento em tempo real, características fundamentais para o desenvolvimento e validação da arquitetura proposta neste trabalho.

3.2.2 Framework Moleculer e Microsserviços disponíveis

O Moleculer (MOLECULER, 2026) é uma plataforma de desenvolvimento baseada em microsserviços que utiliza a linguagem de programação *JavaScript*. Com esse *framework*, é possível criar microsserviços em *Node.js* e permitir que eles acessem os dispositivos periféricos onde estão instalados, disponibilizando-os em uma estrutura de rede.

A Figura 3 apresenta a arquitetura MOA do *framework* Moleculer, na qual os serviços são executados em servidores (nós) individuais que se comunicam via microsserviço de *transporter*, hexágono laranja. Cada microsserviço disponibilizado na arquitetura é representado com um hexágono azul. Os microsserviços podem oferecer e executar diferentes tarefas que são chamadas de ações. O hexágono verde representa o microsserviço de gateway (API *Gateway*), o qual tem a função de interface de conexão dos serviços internos (hexágonos azuis) com aplicações externas através de chamadas via rede ou nuvem por meio do protocolo REST.

Figura 3 – Arquitetura Orientada a Microsserviços do Moleculer



Fonte: (MOLECULER, 2026)

Esse framework foi usado como base para a criação de uma arquitetura de microsserviços para aplicações de automação e controle na I4.0 (PONTAROLLI et al., 2023). Essa arquitetura disponibiliza diversos microsserviços que serão utilizados nesse projeto.

O Microsserviço de Aquisição de dados (DAQ) é responsável pela aquisição de dados de processo e possui uma ação de leitura de entradas, utilizada para a aquisição dos valores medidos pelos sensores, e uma ação para atualizar as saídas, que é usada para ajustar os valores de comando dos inversores de frequência das bombas. O microsserviço PID4.0 foi desenvolvido para aplicações de controle via rede redundantes e é responsável por controlar as variáveis de processo da planta.

3.2.3 Hardware

Na arquitetura de microsserviços da planta piloto, os serviços distribuídos são executados em computadores de placa única do modelo Raspberry Pi 3B+, apresentados na Figura 4. Esse hardware possui conectividade wireless compatível com o padrão IEEE 802.11ac, processador ARM Cortex-A53 de quatro núcleos com arquitetura de 64 bits operando a 1,4 GHz e 1 GB de memória RAM (RASPBERRYPI, 2026).

Para viabilizar a aquisição de dados e o acionamento dos dispositivos de campo, as placas Raspberry Pi são utilizadas em conjunto com a placa de expansão industrial MegaIO, da Sequent Microsystems, conectada na forma de shield (PONTAROLLI et al., 2020). Essa interface permite a leitura de sinais analógicos de sensores no padrão de corrente de 4 a 20 mA, bem como a geração de sinais analógicos de 0 a 10 V destinados ao acionamento dos inversores de frequência das bombas e da válvula proporcional.

Figura 4 – Placa Raspberry Pi 3B+lado esquerdoe MegaIOIndustrialao lado direito



Fonte: (PONTAROLLI et al., 2020)

3.2.4 Node-RED

O Node-RED é uma plataforma de desenvolvimento visual de código aberto amplamente utilizada em aplicações de Internet das Coisas (IoT) e Indústria 4.0, destacando-se pela facilidade de integração entre dispositivos, serviços e protocolos de comunicação. A ferramenta utiliza uma abordagem baseada em fluxos, permitindo a construção de aplicações por meio da conexão de blocos funcionais denominados nós, os quais podem ser configurados através de uma interface gráfica (NODE-RED, 2026).

Essa abordagem simplifica o desenvolvimento de aplicações distribuídas e facilita a implementação de rotinas de aquisição, processamento e encaminhamento de dados em arquiteturas baseadas em microsserviços. Além disso, o Node-RED possui suporte nativo para diferentes protocolos e serviços utilizados em ambientes industriais e aplicações IoT, favorecendo sua integração com sistemas heterogêneos (LARRINAGA et al., 2022).

Neste trabalho, o Node-RED será empregado como orquestrador da infraestrutura de monitoramento, sendo responsável pela aquisição dos dados disponibilizados pelos microsserviços da planta piloto, processamento das informações e envio dos dados ao banco de dados. Dessa forma, a ferramenta atuará como elemento central no fluxo de armazenamento das variáveis de processo e das métricas coletadas da infraestrutura computacional.

3.2.5 InfluxDB

O InfluxDB foi escolhido como o banco de dados utilizado neste trabalho por ser um sistema de gerenciamento de banco de dados orientado a séries temporais, desenvolvido especificamente para o armazenamento, processamento e análise de dados indexados temporalmente, como medições de sensores, métricas computacionais, eventos e variáveis de processo. Diferentemente dos bancos de dados relacionais tradicionais, o InfluxDB é otimizado para aplicações que exigem altas taxas de escrita, consultas eficientes sobre intervalos de tempo e armazenamento contínuo de dados históricos, tornando-se uma solução adequada para aplicações de monitoramento e historiamento em ambientes industriais (CHIABURU, 2026).

Seu modelo de dados é estruturado em medições (*measurements*), campos (*fields*), *tags* e *timestamps*. Os *fields* representam os valores efetivamente armazenados, enquanto as *tags* correspondem a metadados indexáveis utilizados para categorização e filtragem eficiente das informações. O uso de *timestamps* associados a cada registro permite a realização de consultas temporais e análises históricas de forma otimizada (INFLUXDATA, 2026).

Além disso, o InfluxDB oferece mecanismos de compressão de dados e linguagens próprias de consulta, como InfluxQL e Flux, possibilitando operações voltadas à análise de séries temporais. A plataforma também possui integração nativa com ferramentas de monitoramento e visualização, como Grafana e Node-RED, sendo amplamente empregada em aplicações de IoT, supervisão industrial e monitoramento de infraestrutura.

3.2.6 Telegraf

O Telegraf é um agente de coleta de métricas de código aberto desenvolvido pela InfluxData como parte do ecossistema TICK, composto pelas ferramentas Telegraf, InfluxDB, Chronograf e Kapacitor (INFLUXDATA, 2026). A ferramenta é voltada à coleta, processamento e envio de dados de séries temporais para diferentes plataformas de armazenamento e monitoramento, sendo amplamente utilizada em aplicações de infraestrutura, IoT e sistemas industriais.

Sua arquitetura é baseada em plugins modulares, organizados em quatro categorias principais: *inputs*, responsáveis pela aquisição de dados provenientes de sistemas operacionais, sensores, APIs e protocolos industriais; *processors*, utilizados para transformação e tratamento das informações coletadas; *aggregators*, destinados à agregação dos dados antes do

armazenamento; e *outputs*, responsáveis pelo envio das informações para bancos de dados e serviços externos, como o InfluxDB.

Essa abordagem modular proporciona flexibilidade na integração de diferentes fontes de dados e facilita sua utilização em arquiteturas distribuídas e ambientes de monitoramento em tempo real. Além disso, o Telegraf apresenta baixo consumo de recursos computacionais e configuração simplificada por meio de arquivos textuais, características que favorecem sua utilização em dispositivos de Computação em Borda, como as Raspberry Pi empregadas neste trabalho. Neste projeto, o Telegraf será utilizado para o monitoramento das métricas de hardware dos dispositivos responsáveis pela execução da arquitetura PnPr.

3.2.7 Grafana

O Grafana é uma plataforma de visualização e análise de dados de código aberto amplamente utilizada em aplicações de monitoramento de sistemas computacionais, infraestrutura, processos industriais e IoT (GRAFANA, 2026). A ferramenta possibilita a criação de *dashboards* interativos para apresentação de métricas em tempo real e análise histórica de dados armazenados em diferentes fontes.

A plataforma possui compatibilidade com diversos bancos de dados e sistemas de monitoramento, como InfluxDB, Prometheus, MySQL e PostgreSQL, permitindo a integração de múltiplas fontes de informação em uma única interface de visualização. Por meio de painéis configuráveis, é possível apresentar gráficos, tabelas, indicadores e alertas de forma dinâmica, facilitando o acompanhamento do comportamento das variáveis monitoradas e a identificação de padrões e anomalias operacionais. Neste trabalho, o Grafana será utilizado para o desenvolvimento dos *dashboards* responsáveis pela visualização em tempo real e pela análise histórica das variáveis de processo e das métricas de hardware.

3.3 Métodos

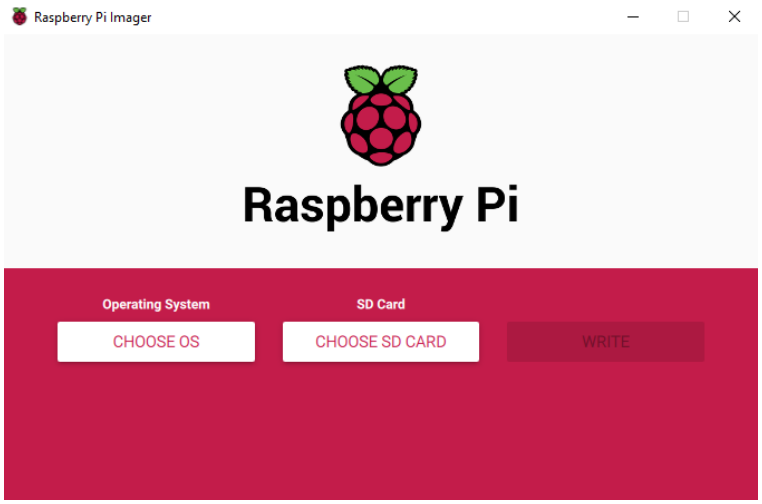
Conforme apresentado na Seção 3.2.3, a infraestrutura da planta piloto é composta por múltiplas placas Raspberry Pi interconectadas em rede, formando um ambiente distribuído compatível com os princípios da Computação em Borda. Para o desenvolvimento deste trabalho, uma dessas unidades foi destinada à execução da infraestrutura de monitoramento proposta, concentrando as aplicações responsáveis pela aquisição das variáveis de processo,

armazenamento das séries temporais, monitoramento das métricas de hardware e visualização dos dados.

Considerando a necessidade de armazenamento contínuo das informações geradas pelo sistema ao longo de extensos períodos de operação, essa Raspberry Pi foi equipada com uma unidade de armazenamento *Solid State Drive* (SSD) de 480 GB, substituindo o cartão *microSD* convencional de 32 GB. Essa solução proporciona maior capacidade para o armazenamento do banco de dados InfluxDB, além de aumentar a confiabilidade da infraestrutura para aplicações que demandam registro histórico de dados por longos períodos.

Inicialmente, o dispositivo foi configurado utilizando a ferramenta Raspberry Pi *Imager*, responsável pela gravação do sistema operacional. Foi instalada uma distribuição Linux com interface gráfica utilizando a unidade SSD como meio de armazenamento. A Figura 5 apresenta a ferramenta utilizada durante esse processo de configuração.

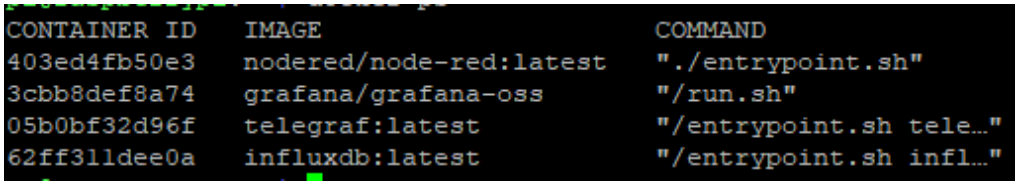
Figura 5 – Raspberry Pi imager



fonte: (Raspberry Pi 2026)

Após a configuração do ambiente operacional, cada aplicação foi implantada em um contêiner independente, incluindo o Node-RED, InfluxDB, Grafana e Telegraf, conforme ilustrado na Figura 6. Essa abordagem permitiu o isolamento dos serviços, facilitando atividades de configuração, manutenção, atualização e diagnóstico ao longo do desenvolvimento do projeto.

Figura 6 – Containers instalados



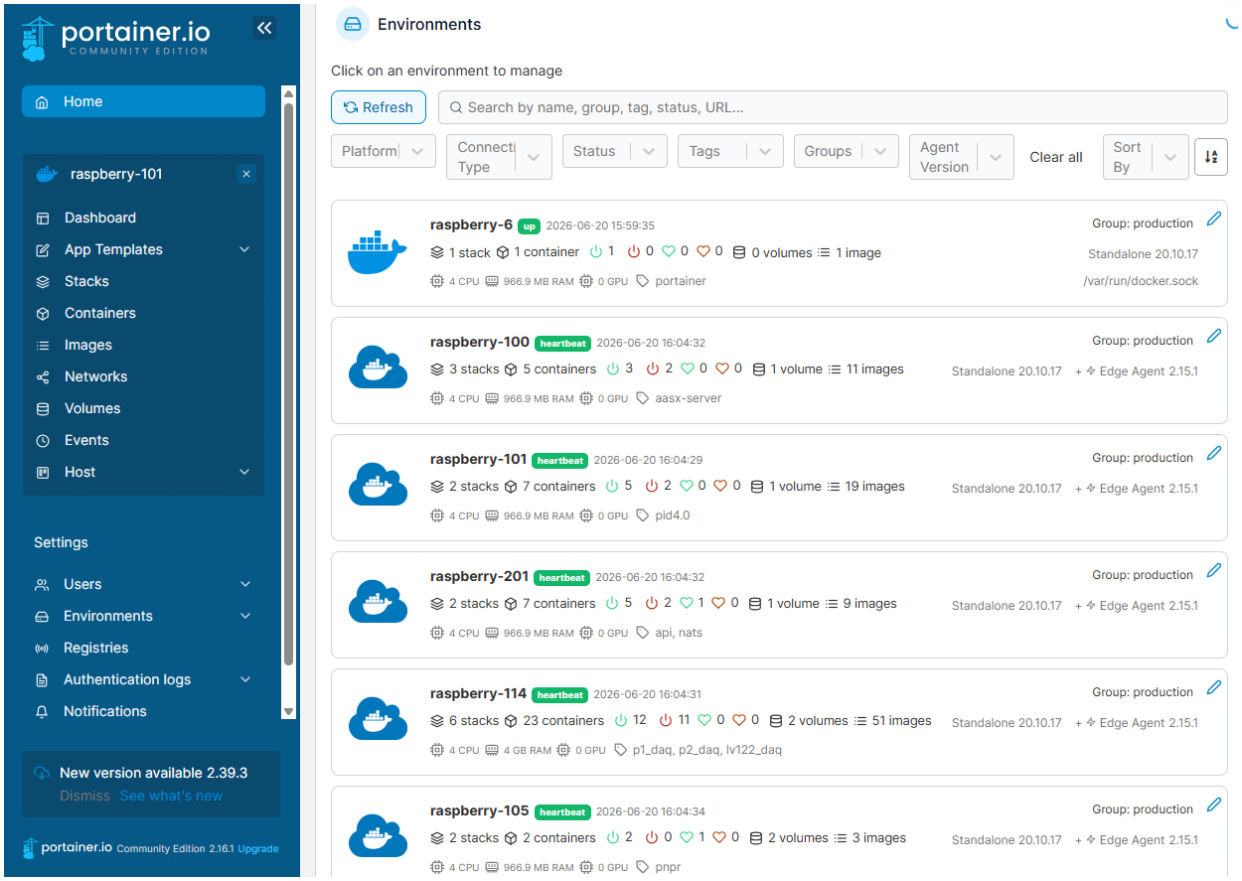
Fonte: Elaborado pelo autor

A utilização de contêineres também contribui para a padronização da infraestrutura computacional, garantindo que as aplicações sejam executadas sob as mesmas condições independentemente do hardware utilizado. Além disso, essa estratégia facilita a identificação e correção de falhas, uma vez que cada serviço pode ser monitorado e reiniciado individualmente sem impactar os demais componentes da arquitetura.

Outro benefício relevante está relacionado à portabilidade da solução. Como todas as dependências e configurações necessárias para a execução dos serviços encontram-se encapsuladas nos contêineres, a replicação da arquitetura em outros dispositivos torna-se mais simples, contribuindo para a escalabilidade e reprodutibilidade da infraestrutura desenvolvida neste trabalho.

Essa padronização está alinhada aos princípios da arquitetura Pnpr implementada na planta piloto, na qual os serviços são distribuídos e gerenciados de forma modular. Além da execução dos contêineres, a infraestrutura utiliza a ferramenta Portainer para o gerenciamento centralizado dos dispositivos da rede. Por meio dessa plataforma, é possível monitorar as Raspberry Pi conectadas, bem como os contêineres, volumes e imagens docker presentes em cada dispositivo. A Figura 7 apresenta a interface de gerenciamento disponibilizada pela ferramenta.

Figura 7 – Interface de gerenciamento Portainer.io



Fonte: Elaborado pelo autor

Na Raspberry Pi dedicada ao monitoramento são executados o orquestrador Node-RED, o banco de dados InfluxDB, o Grafana e o agente Telegraf responsável pela coleta das métricas locais de hardware. Nas demais Raspberry Pi da planta são executados apenas os agentes coletores de métricas, responsáveis pelo monitoramento distribuído dos dispositivos. Essa organização integra os dois eixos de monitoramento propostos neste trabalho, permitindo o armazenamento histórico das variáveis de processo provenientes dos *assets* da planta e o acompanhamento contínuo do desempenho computacional da infraestrutura.

4 DESENVOLVIMENTO

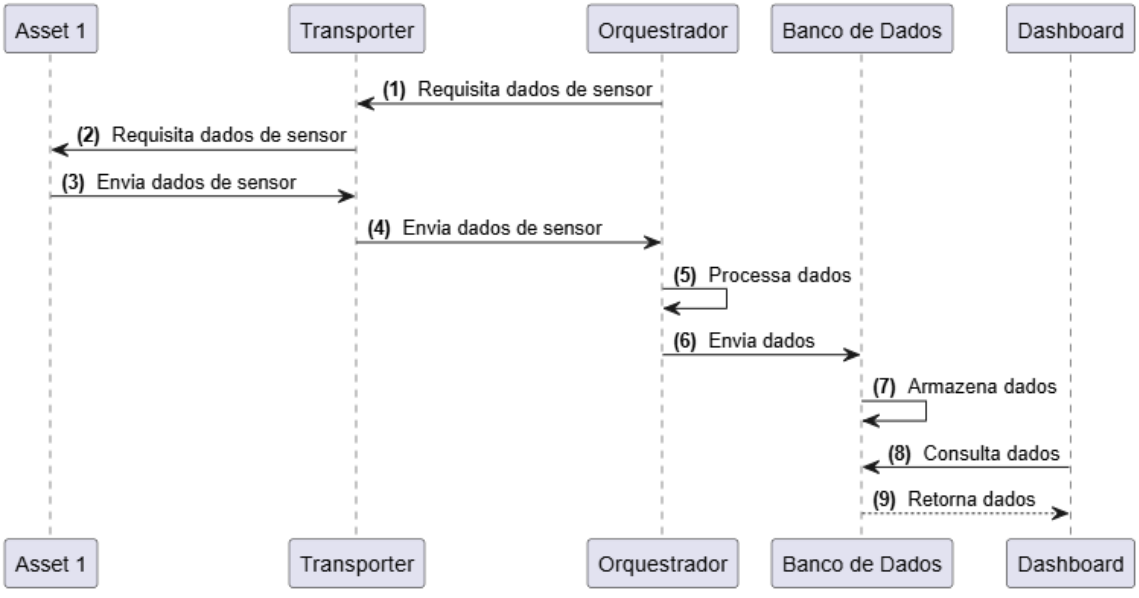
Esta seção apresenta o desenvolvimento da infraestrutura de monitoramento proposta neste trabalho. Inicialmente, são descritos os fluxos de comunicação dos dois eixos que compõem a solução: o monitoramento das variáveis de processo e o monitoramento das métricas de hardware, detalhando como ocorre a aquisição, o processamento e o armazenamento das informações em cada caso.

Em seguida, são apresentadas as etapas de implementação da infraestrutura, abrangendo a configuração do banco de dados InfluxDB, o desenvolvimento da orquestração no Node-RED, a configuração do agente Telegraf, a construção dos *dashboards* no Grafana e a integração da solução à arquitetura *Plug-and-Produce*.

4.1 Fluxo de monitoramento das variáveis de processo

O fluxo de monitoramento das variáveis de processo inicia-se com a requisição das informações disponibilizadas pelos *assets* da planta industrial. Conforme ilustrado na Figura 8, o orquestrador da camada de monitoramento comunica-se com os microsserviços da arquitetura por meio do *transporter*, solicitando periodicamente os valores das variáveis de interesse.

Figura 8 – Diagrama de sequência Monitoramento das Variáveis de Processo



Fonte: Elaborado pelo autor

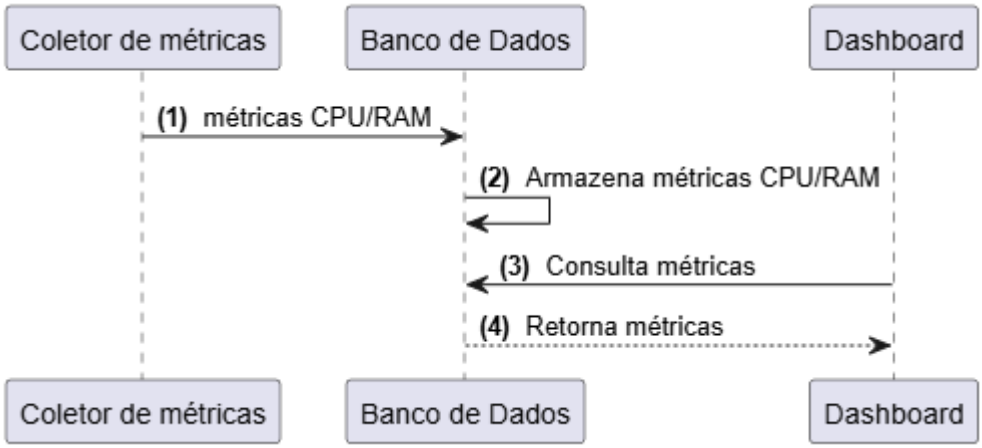
Após o processamento da requisição pelo microsserviço correspondente, a resposta é enviada ao orquestrador, que recebe os dados, realiza o processamento necessário e os estrutura no formato exigido pelo banco de dados de séries temporais. Nessa etapa, também são definidos os respectivos *measurements*, *fields* e *tags* utilizados para identificação e organização das informações.

Em seguida, os dados são armazenados no banco de dados juntamente com suas marcações temporais, possibilitando o registro contínuo do comportamento das variáveis monitoradas. A partir das informações armazenadas, são construídos *dashboards* responsáveis por consultar os dados registrados e disponibilizá-los de forma gráfica, constituindo a camada de monitoramento da planta piloto.

4.2 Fluxo de monitoramento das métricas de hardware

O fluxo de monitoramento das métricas de hardware inicia-se com a coleta periódica das informações de desempenho dos dispositivos que compõem a arquitetura distribuída. Conforme apresentado na Figura 9, essa aquisição é realizada por agentes coletores instalados nas placas Raspberry Pi, responsáveis por monitorar continuamente recursos como utilização de CPU, consumo de memória RAM, utilização de disco e temperatura do processador.

Figura 9 – Diagrama de sequência Monitoramento das Métricas de Hardware



Fonte: Elaborado pelo autor

Após a coleta, os agentes estruturam as informações e as enviam diretamente ao banco de dados de séries temporais. Durante esse processo, cada registro é armazenado

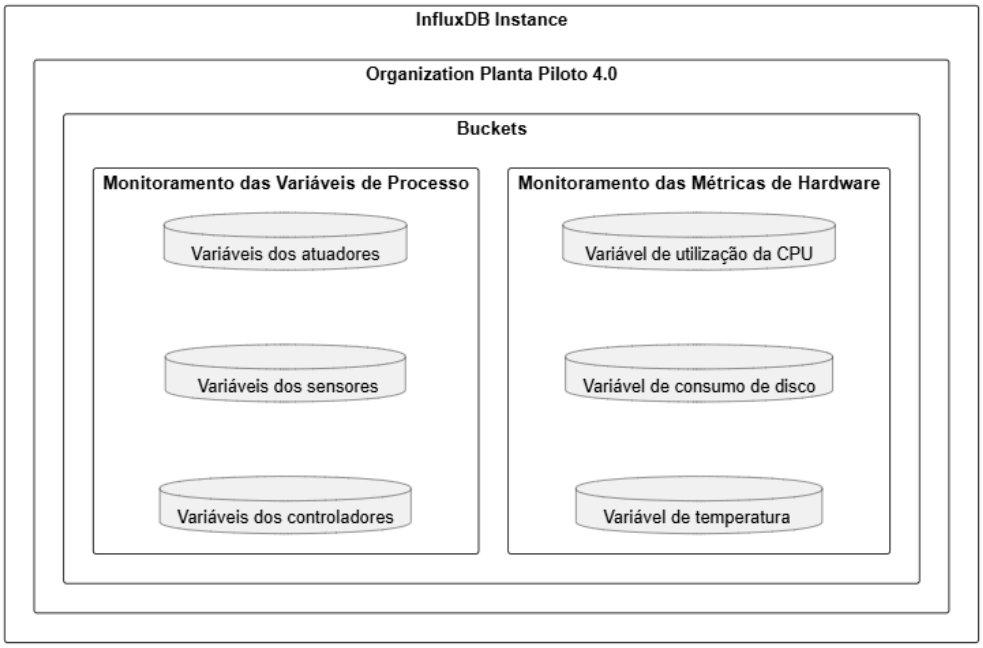
juntamente com suas marcações temporais e metadados de identificação, permitindo distinguir tanto o tipo de métrica quanto o dispositivo de origem. Posteriormente, as métricas de hardware, assim como as variáveis de processo, são disponibilizadas em *dashboards*, permitindo o monitoramento em tempo real e a análise histórica da infraestrutura desenvolvida.

4.3 Implementação do Banco de Dados

Com a infraestrutura devidamente instalada, inicia-se a configuração do banco de dados InfluxDB, responsável pelo armazenamento das séries temporais geradas pela arquitetura. Antes da aquisição das variáveis de processo e das métricas de hardware, é necessário definir a organização das informações no banco de dados. Para isso, em conformidade com os dois eixos de monitoramento estabelecidos neste trabalho, foram criados dois *buckets* distintos, destinados ao armazenamento das variáveis de processo e das métricas de hardware, respectivamente.

O primeiro *bucket* foi destinado ao armazenamento das variáveis de processo da planta piloto, enquanto o segundo foi configurado para armazenar as métricas de hardware coletadas dos dispositivos que compõem a arquitetura. Essa organização permite separar logicamente os diferentes tipos de dados monitorados, facilitando sua consulta, manutenção e utilização na construção dos *dashboards* de supervisão. A Figura 10 apresenta a estrutura adotada para o banco de dados. Conforme apresentado nos diagramas das Figuras 8 e 9, após a configuração do banco de dados, os agentes responsáveis pela aquisição das informações passam a realizar a escrita dos dados nos respectivos *buckets*.

Figura 10 – Estrutura banco de dados



Fonte: Elaborado pelo autor

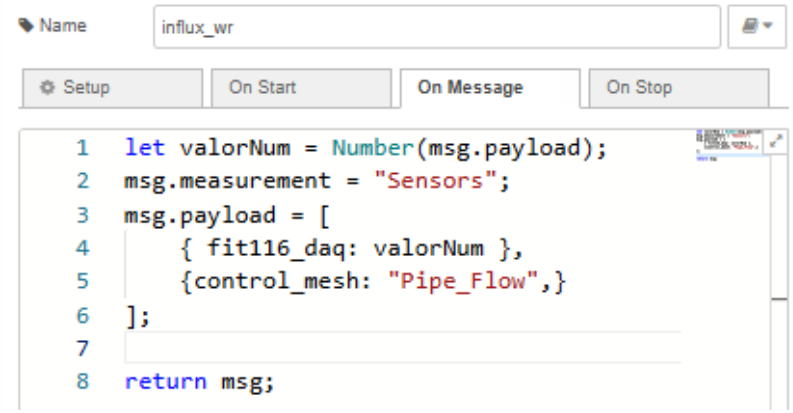
4.4 Implementação da Orquestração

Para realizar a orquestração, isto é, a requisição das variáveis de processo geradas pelos microsserviços e sua escrita no banco de dados, utiliza-se a ferramenta Node-RED. Por meio da biblioteca Moleculer, é possível realizar as requisições diretamente utilizando o *transport* NATS, dispensando a necessidade de um API *Gateway*. Essa abordagem permite que a comunicação ocorra diretamente pela infraestrutura de mensageria da arquitetura, eliminando etapas intermediárias associadas ao *gateway* de aplicação e reduzindo a sobrecarga de comunicação entre os componentes distribuídos.

Após o envio da requisição pelo orquestrador, o microsserviço responsável retorna o valor solicitado por meio do *Transporter*. Em seguida, a resposta recebida e processada no Node-RED, onde os dados são convertidos para formatos compatíveis com o modelo de armazenamento do InfluxDB. Nessa etapa, também são definidas as respectivas *measurement*, *tags* e *filds* de identificação associadas a cada variável monitorada, permitindo sua correta organização e posterior consulta no banco de dados. A Figura 11 apresenta o processamento realizado após o recebimento da resposta do microsserviço. Já a Tabela 1 relaciona as variáveis

monitoradas, bem como seus respectivos metadados utilizados na estruturação dos dados no InfluxDB.

Figura 11 – Processamento da resposta Microsserviço



Fonte: Elaborado pelo autor

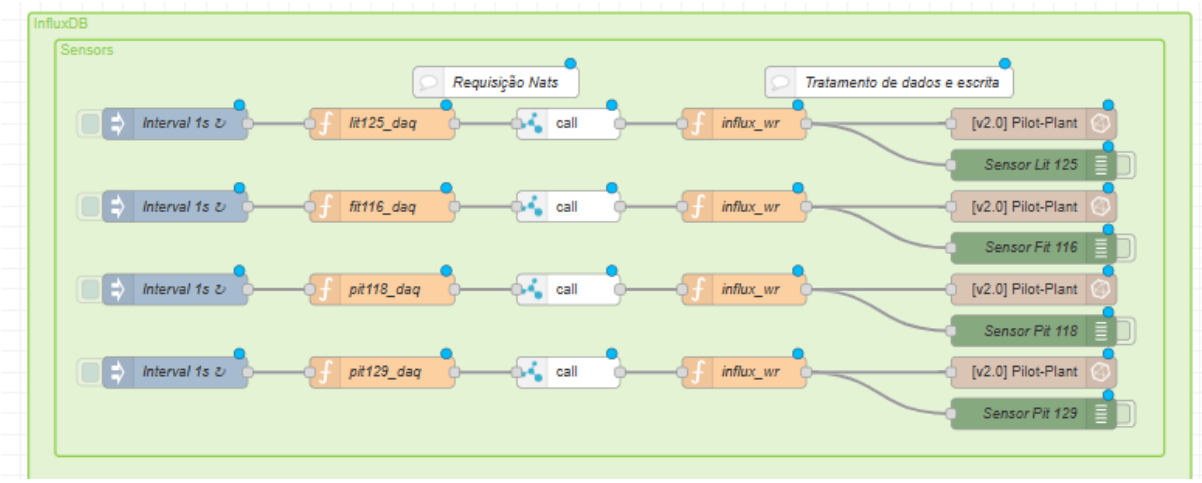
Tabela 1 – Variáveis de processo no InfluxDB

Microsserviço	Measurement	Field	Tag (control_mesh)
lit125_daq	Sensors	lit125_daq	Tank_Level
fit116_daq	Sensors	fit116_daq	Pipe_Flow
pit118_daq	Sensors	pit118_daq	Pipe_Pressure
pit129_daq	Sensors	pit129_daq	Reservoir-pressure
lic125_pid4	controller_data	erro	Tank_Level
lic125_pid4	controller_data	setpoint	Tank_Level
fic116_pid4	controller_data	erro	Pipe_Flow
fic116_pid4	controller_data	setpoint	Pipe_Flow
pic118_pid4	controller_data	erro	Pipe_Pressure
pic118_pid4	controller_data	setpoint	Pipe_Pressure
pic129_pid4	controller_data	erro	Reservoir-pressure
pic129_pid4	controller_data	setpoint	Reservoir-pressure
p1_daq	Actuators	p1_daq	-
P2_daq	Actuators	p2_daq	-
lv122_daq	Actuators	lv122_daq	-

Fonte: Elaborado pelo autor

Concluído o processamento, os dados são enviados ao InfluxDB por meio do nó específico disponibilizado pela biblioteca da plataforma para o Node-RED. Para isso, são configurados parâmetros como endereço do servidor, organização e *bucket* de destino, responsáveis por definir o local de armazenamento das informações. A Figura 12 apresenta o fluxo completo de orquestração desenvolvido para a aquisição, processamento e armazenamento das variáveis de processo da planta piloto.

Figura 12 – Flow de orquestração Node-RED



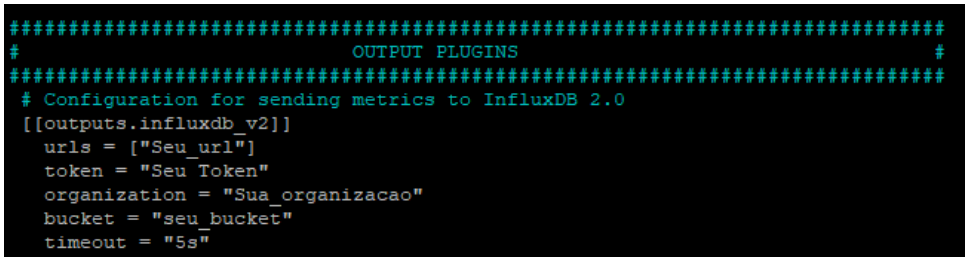
Fonte: Elaborado pelo autor

4.5 Implementação do Monitoramento de Hardware

Após a instalação do Telegraf em um contêiner docker, foi necessária a configuração de seu arquivo de parâmetros para definir quais métricas de hardware seriam coletadas e para qual destino essas informações seriam enviadas. Entre as métricas monitoradas estão a utilização da CPU, consumo de memória RAM, utilização de disco e temperatura dos dispositivos.

A coleta dessas informações é realizada por meio dos plugins de entrada disponibilizados pelo Telegraf, enquanto o envio dos dados ao banco de dados é configurado por meio dos plugins de saída. Neste trabalho, foi utilizado o plugin influxdb_v2, responsável por estabelecer a comunicação com o InfluxDB e realizar a escrita dos dados diretamente no *bucket* destinado ao armazenamento das métricas de hardware. O trecho de configuração apresentado a seguir ilustra os parâmetros utilizados para conexão com o banco de dados.

Figura 13 – Plugin de conexão Telegraf com o Node-RED



Fonte: Elaborado pelo autor

Além da Raspberry Pi responsável pela execução central da arquitetura, o Telegraf também foi instalado em uma segunda Raspberry Pi pertencente à infraestrutura da planta piloto. Esse procedimento teve como objetivo validar a capacidade de monitoramento distribuído proposta neste trabalho.

Uma das vantagens da utilização do Telegraf é a facilidade de replicação da configuração para novos dispositivos. Após a configuração inicial do agente em uma Raspberry Pi, o mesmo arquivo de configuração pode ser copiado para os demais dispositivos da rede, sendo necessária apenas a alteração do identificador associado ao equipamento monitorado. Dessa forma, torna-se possível expandir o sistema de monitoramento para toda a infraestrutura computacional da planta com reduzido esforço de configuração.

As métricas coletadas pelo Telegraf foram armazenadas no InfluxDB utilizando *measurements* e *tags* que permitem identificar tanto o tipo de métrica quanto o dispositivo de origem. A Tabela 2 apresenta as principais informações monitoradas e armazenadas no banco de dados.

Tabela 2 – Métricas de hardware utilizadas

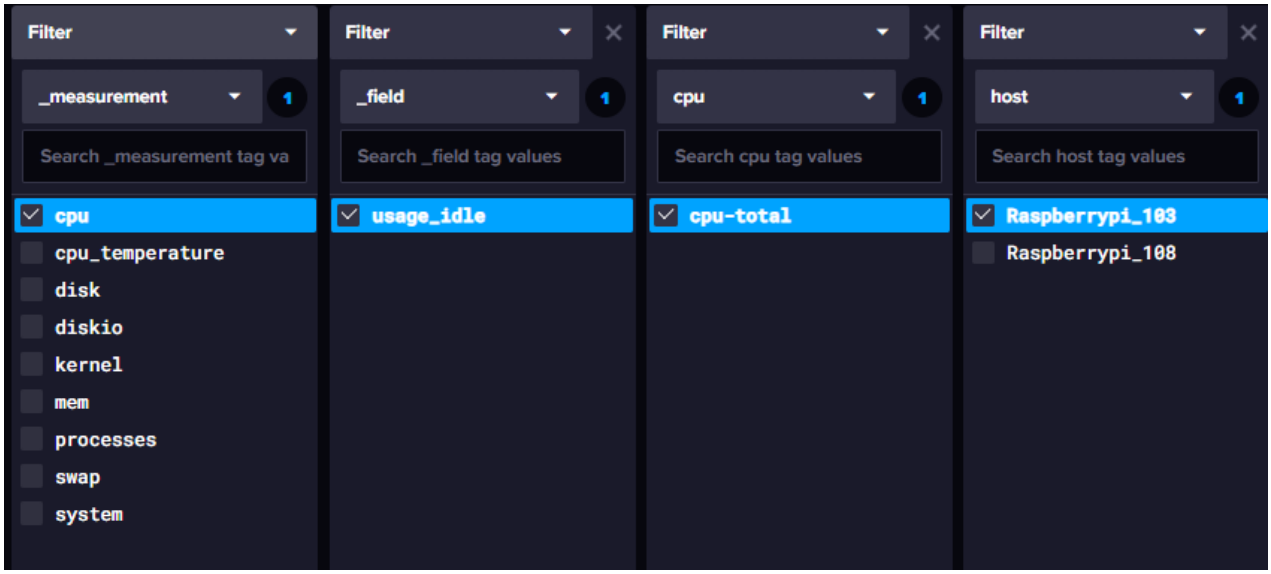
Measurement	Field	Tag	Descrição
cpu	usage_idle	host/rasp108	Percentual de CPU ociosa raspberrypi 108
cpu	usage_idle	host/rasp103	Percentual de CPU ociosa raspberrypi 103
mem	used_percent	host/rasp108	Porcentagem de memória RAM utilizada raspberrypi 103
mem	used_percent	host/rasp103	Porcentagem de memória RAM utilizada raspberrypi 103
disk	total	host/rasp108	Espeço total de disco raspberrypi 108
disk	total	host/rasp103	Espeço total de disco raspberrypi 103
disk	used	host/rasp108	Espeço usado de disco raspberrypi 108

disk	used	host/rasp103	Espeço usado de disco raspberrypi 103
Cpu_temperature	-	host/rasp108	Temperatura do CPU raspberrypi 108
Cpu_temperature	-	host/rasp103	Temperatura do CPU raspberrypi 103

Fonte: Elaborado pelo autor

A Figura 14 apresenta os dados provenientes das diferentes Raspberry Pi sendo armazenados no *bucket* de métricas de hardware do InfluxDB. A utilização de identificadores distintos permite diferenciar a origem de cada conjunto de dados, possibilitando o monitoramento individualizado dos dispositivos e a comparação de seu desempenho ao longo do tempo.

Figura 14 – Escrita das métricas de hardware no InfluxDB



Fonte: Elaborado pelo autor

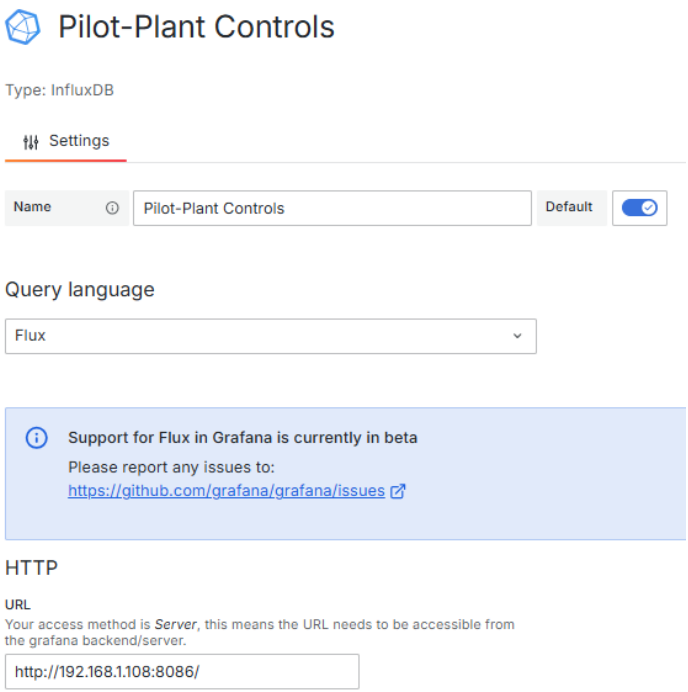
4.6 Desenvolvimento dos Dashboards

Após a configuração do InfluxDB, da orquestração desenvolvida no Node-RED e coleta de métricas dos agentes Telegraf, todos os dados referentes às variáveis de processo e às métricas de hardware passaram a ser armazenados continuamente no banco de dados. Com essa infraestrutura operacional, tornou-se possível utilizar o Grafana para consultar as informações registradas e construir *dashboards* destinados ao monitoramento em tempo real e à análise histórica da arquitetura.

O primeiro passo para a criação dos *dashboards* consistiu na configuração da fonte de dados (*Data Source*) no Grafana, estabelecendo a comunicação entre a plataforma de visualização e o banco de dados InfluxDB. Para isso, foi adicionada uma nova fonte de dados do tipo InfluxDB, na qual foram informados o endereço do servidor, a organização (*Organization*), o *token* de autenticação e o *bucket* contendo as séries temporais a serem consultadas.

Após o preenchimento desses parâmetros, o Grafana realiza a autenticação junto ao InfluxDB e passa a ter acesso às informações armazenadas no banco de dados. A partir dessa conexão, torna-se possível executar consultas utilizando a linguagem Flux, recuperar os dados registrados e utilizá-los na construção de painéis gráficos, indicadores, tabelas e demais componentes de visualização disponibilizados pela plataforma. A Figura 15 apresenta a configuração da fonte de dados utilizada para estabelecer a conexão entre o Grafana e o InfluxDB.

Figura 15 – Conexão Grafana e InfluxDB

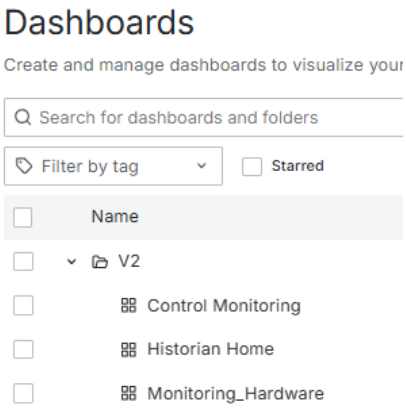


Fonte: Elaborado pelo autor

Com a conexão estabelecida, foi realizada a estruturação dos *dashboards* no Grafana. Para facilitar a organização das informações, o historiador foi dividido em três páginas: *Home*, *Control Monitoring* e *Hardware Monitoring*.

A Figura 16 apresenta as três páginas que compõem o *dashboard* desenvolvido no Grafana: A página Home tem como objetivo apresentar uma visão geral do historiador de processos, descrevendo a finalidade do sistema, as funções de cada página de monitoramento e fornecendo orientações básicas para sua utilização. A página *Control Monitoring* é destinada ao monitoramento das variáveis de processo da planta piloto, enquanto a página *Hardware Monitoring* concentra a visualização das métricas de desempenho dos dispositivos que compõem a infraestrutura computacional. As funcionalidades e os elementos de cada página são detalhados nas subseções a seguir.

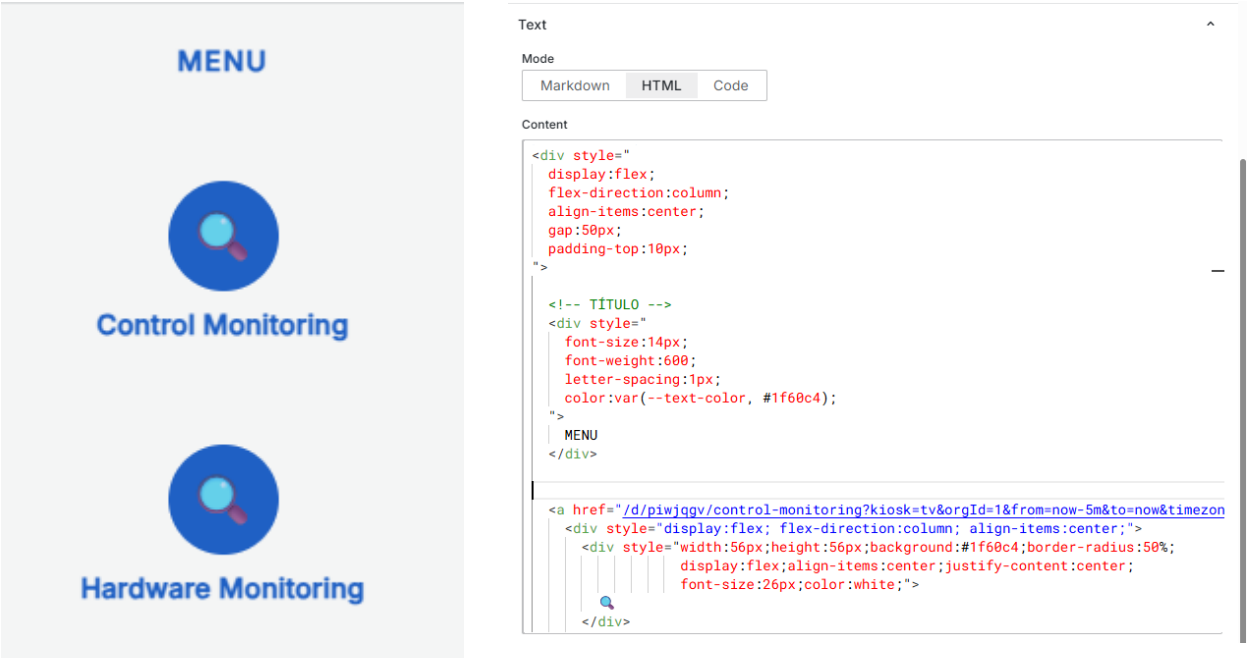
Figura 16 – Páginas Dashboard



Fonte: Elaborado pelo autor

Considerando a divisão do sistema em diferentes páginas, foi implementado em todas elas um menu de navegação para facilitar a utilização dos *dashboards*. Esse menu foi desenvolvido por meio de um painel de texto em HTML, responsável por disponibilizar links que redirecionam o usuário para a página selecionada dentro do Grafana. Dessa forma, é possível alternar rapidamente entre as diferentes áreas de monitoramento sem a necessidade de retornar à tela inicial da plataforma. A Figura 17 apresenta, à esquerda, a interface do menu de navegação incorporado ao *dashboard* e, à direita, o trecho do código HTML utilizado para sua implementação.

Figura 17 – Menu Dashboard



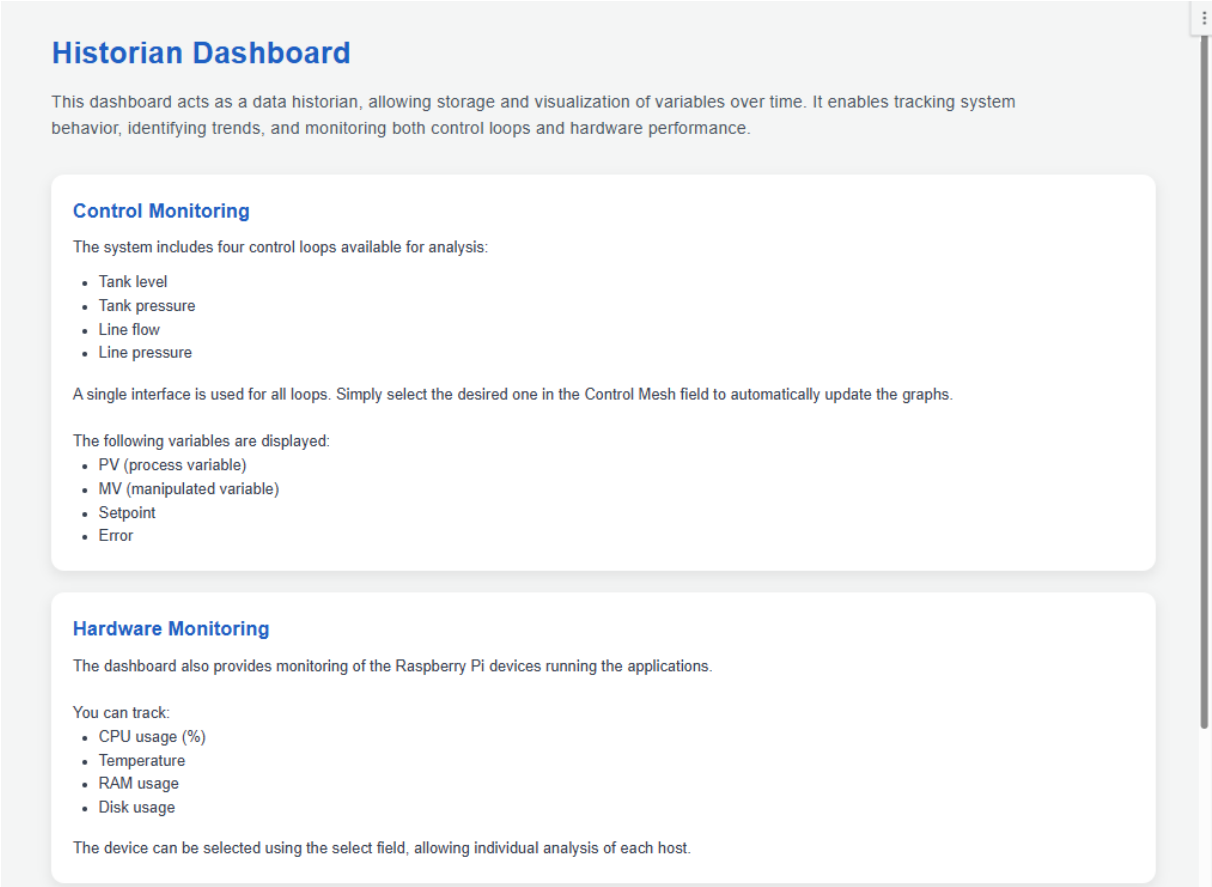
Fonte: Elaborado pelo autor

4.6.1 Dashboard Home

A página *Home* foi desenvolvida para atuar como ponto de partida para a utilização do sistema de monitoramento. Seu objetivo não é apresentar uma descrição detalhada dos conceitos relacionados ao monitoramento de processos e de hardware, mas fornecer informações introdutórias que auxiliem usuários sem experiência prévia na utilização da plataforma. Para isso, a página apresenta uma visão geral do historiador de processos, descreve a finalidade de cada *dashboard* de monitoramento, indica as variáveis monitoradas e fornece instruções básicas para navegação e utilização do sistema.

Com essa finalidade, a página foi estruturada em três blocos de texto: *Control Monitoring*, que apresenta uma breve descrição do painel de monitoramento destinado ao monitoramento das variáveis de processo; *Hardware Monitoring*, que descreve o *dashboard* responsável pelo acompanhamento das métricas de hardware dos dispositivos da infraestrutura; e *How to Use*, que reúne orientações básicas para a utilização da plataforma e navegação entre os dashboards. A Figura 18 apresenta a interface desse dashboard.

Figura 18 – Dashooard Historian Home



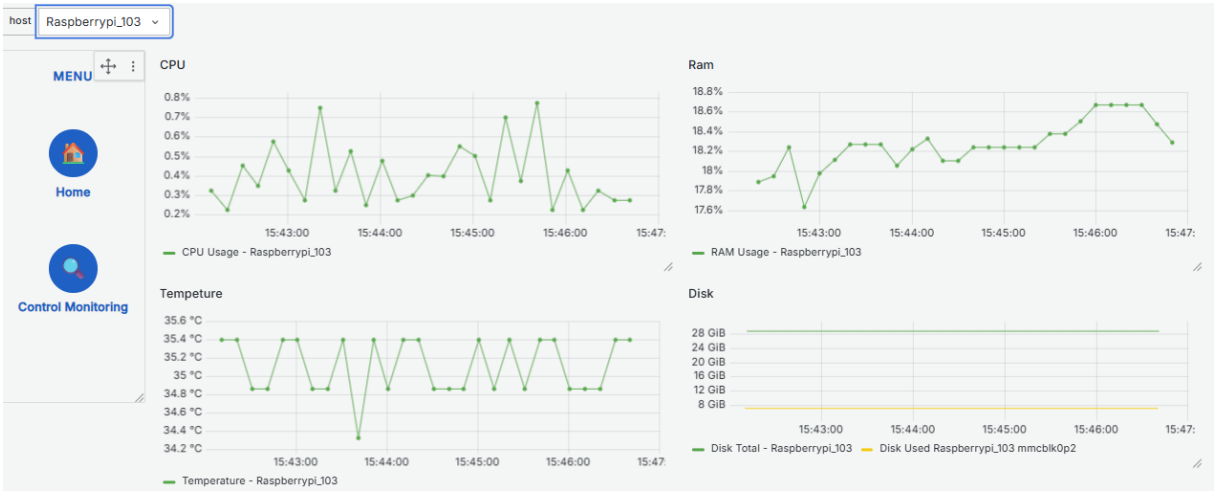
Fonte: Elaborado pelo autor

4.6.2 Dashboard Hardware Monitoring

O *Hardware Monitoring* é composto por quatro gráficos (Figura 19), cada um destinado ao monitoramento de uma métrica específica de hardware, conforme apresentado na Tabela 2. O gráfico localizado na região superior esquerda exibe a utilização da CPU em porcentagem, enquanto o gráfico superior direito apresenta o percentual de utilização da memória RAM. Na região inferior esquerda é monitorada a temperatura do dispositivo, expressa em graus Celsius, e o gráfico inferior direito apresenta a capacidade de armazenamento em disco, indicando tanto o espaço utilizado quanto o espaço total disponível, em *gigabytes*.

Essa organização permite que as principais métricas de desempenho da infraestrutura sejam visualizadas simultaneamente, facilitando o acompanhamento do estado operacional dos dispositivos monitorados e a rápida identificação de possíveis anomalias ou degradações de desempenho.

Figura 19 – Dashboard hardware monitoring



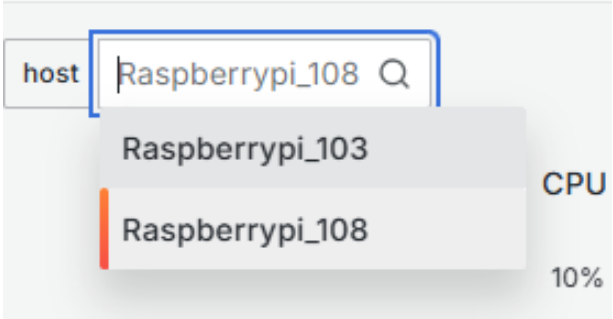
Fonte: Elaborado pelo autor

Considerando que a infraestrutura proposta foi concebida para ser escalável e possibilitar o monitoramento de um número crescente de dispositivos, a criação de um *dashboard* específico para cada equipamento tornaria a solução pouco prática e de difícil manutenção. Dessa forma, optou-se pelo desenvolvimento de um mecanismo que permite utilizar o mesmo painel para monitorar diferentes dispositivos da infraestrutura.

Para isso, foi criada uma variável no Grafana responsável por consultar os valores da *tag host* armazenada no InfluxDB, essa *tag* é responsável por identificar os dispositivos presentes na planta. Como cada métrica de hardware é registrada juntamente com essa *tag*, a seleção de um determinado valor dessa variável faz com que todas as consultas (queries) do *dashboard* sejam filtradas automaticamente, exibindo apenas os dados correspondentes ao equipamento selecionado.

Na interface do Grafana, essa funcionalidade é disponibilizada por meio de uma caixa de seleção posicionada no canto superior esquerdo da tela, permitindo que o usuário alterne rapidamente entre os dispositivos monitorados sem a necessidade de acessar diferentes *dashboards*. A Figura 20 apresenta a caixa de seleção disponibilizada para o usuário.

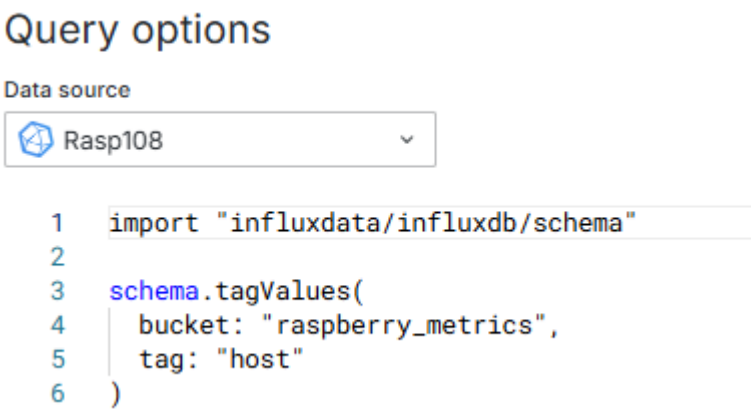
Figura 20 – Seleção do dispositivo monitorado



Fonte: Elaborado pelo autor

A variável criada no Grafana consulta diretamente o banco de dados InfluxDB por meio da fonte de dados configurada. Para isso, é utilizada uma consulta em Flux que emprega a função *schema.tagValues()* (Figura 21), responsável por recuperar todos os valores associados ao *host*. Dessa forma, a variável é preenchida automaticamente com a identificação dos dispositivos monitorados, permitindo que o usuário selecione o equipamento desejado.

Figura 21 – Variável Grafana host



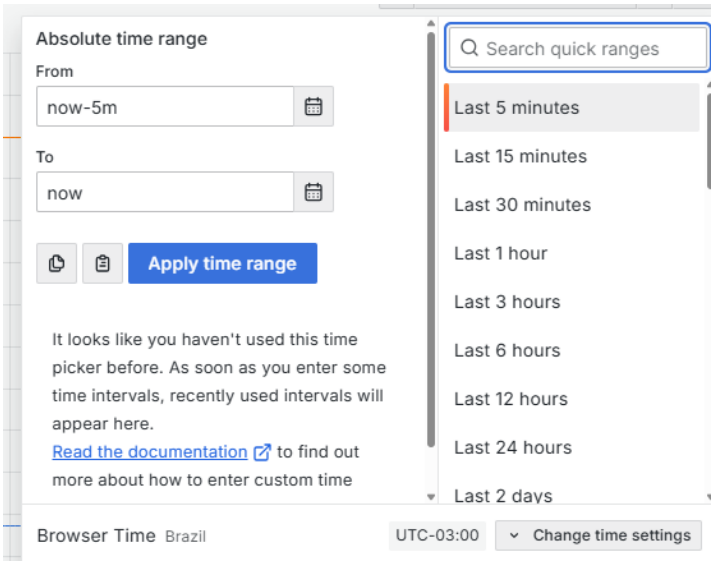
Fonte: Elaborado pelo autor

Uma das principais funcionalidades do Grafana para análise histórica dos dados é o *Time Range*, ilustrado na Figura 22. Esse recurso permite definir o intervalo de tempo utilizado nas consultas ao banco de dados, possibilitando tanto o monitoramento em tempo real quanto a análise de eventos passados.

Por meio dessa funcionalidade, o usuário pode selecionar períodos predefinidos, como os últimos 30 dias, 24 horas ou 15 minutos, bem como especificar um intervalo de datas e horários personalizado. Após a seleção do período desejado, todas as consultas realizadas

pelos painéis do *dashboard* são atualizadas automaticamente, exibindo apenas os dados correspondentes ao intervalo de tempo selecionado.

Figura 22 – Configuração de time range



Fonte: Elaborado pelo autor

A possibilidade de monitorar períodos de até um ano gera um desafio relacionado ao volume de dados armazenados. Como o sistema registra uma nova amostra a cada 10 segundo para o monitoramento de hardware e 1 segundo para o monitoramento da planta, consultas em janelas de tempo muito extensas podem resultar na recuperação de milhões de pontos. Além de aumentar significativamente o tempo de processamento da consulta, esse volume de dados pode exceder a capacidade de renderização do Grafana, causando lentidão ou até mesmo falhas na exibição dos gráficos.

Para contornar esse problema, foi implementada uma estratégia de agregação dinâmica dos dados, aplicada em todas as consultas responsáveis pela geração dos gráficos. Essa estratégia ajusta automaticamente a resolução temporal da consulta de acordo com o intervalo de tempo selecionado pelo usuário, reduzindo a quantidade de pontos retornados sem comprometer a representatividade das informações. Conforme ilustrado na Figura 23, o sistema define o intervalo de agregação (*window*) da seguinte forma:

- Quando o *windowPeriod* é de até 1 segundo, os dados são agregados em intervalos de 1 segundo;

- Quando o *windowPeriod* é superior a 1 segundo e de até 10 segundos, os dados são agregados em intervalos de 10 segundos;
- Quando o *windowPeriod* é superior a 10 segundos e de até 1 minuto, a agregação é realizada em intervalos de 1 minuto;
- Quando o *windowPeriod* é superior a 1 minuto, os dados são agregados em intervalos de 30 minutos.

Em seguida, a função *aggregateWindow()* calcula a média dos valores dentro de cada intervalo definido, reduzindo significativamente o número de pontos enviados ao painel. Essa abordagem melhora o desempenho das consultas, diminui o consumo de recursos e garante a visualização eficiente dos dados, mesmo para períodos extensos de monitoramento.

Figura 23 – Requisição de uso de CPU

A (Pilot-Plant)

```
windowNs = int(v: v.windowPeriod)

window =
    if windowNs <= int(v: 1s) then 1s
    else if windowNs <= int(v: 10s) then 10s
    else if windowNs <= int(v: 1m) then 1m
    else 30m

from(bucket: "raspberry_metrics")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r._measurement == "cpu")
  |> filter(fn: (r) => r._field == "usage_idle")
  |> filter(fn: (r) => r.cpu == "cpu-total")
  |> filter(fn: (r) => r.host == "${host}")
  |> aggregateWindow(every: window, fn: mean, createEmpty: false)
  |> map(fn: (r) => ({
      r with
      _value: 100.0 - r._value,
      _field: "CPU Usage - " + r.host
  }))
  |> keep(columns: ["_time", "_value", "_field"])
```

Flux language syntax | Sample query

Fonte: Elaborado pelo autor

4.6.3 Dashboard Control Monitoring

O *Control Monitoring* foi desenvolvido para possibilitar o acompanhamento das principais variáveis envolvidas nas malhas de controle da planta piloto (Figura 24). Seu elemento central é um gráfico temporal que apresenta, simultaneamente, quatro variáveis provenientes dos microsserviços de controle: a variável de processo (*Process Variab*le – PV), o valor de referência (*Setpoint* – SP), a variável manipulada (*Manipulated Variab*le – MV) e o

erro de controle. Como todas essas variáveis são registradas em porcentagem, optou-se por consolidá-las em um único gráfico, permitindo a comparação direta de seu comportamento ao longo do tempo e facilitando a análise da dinâmica da malha de controle.

As consultas do *Control Monitoring* utilizam a mesma estratégia de agregação dinâmica apresentada na Seção 4.4.2 *Hardware Monitoring*, na qual a variável *window* define automaticamente o intervalo de agregação empregado pela função *aggregateWindow().* Essa abordagem permite manter o desempenho das consultas e da visualização dos gráficos mesmo para grandes janelas de tempo.

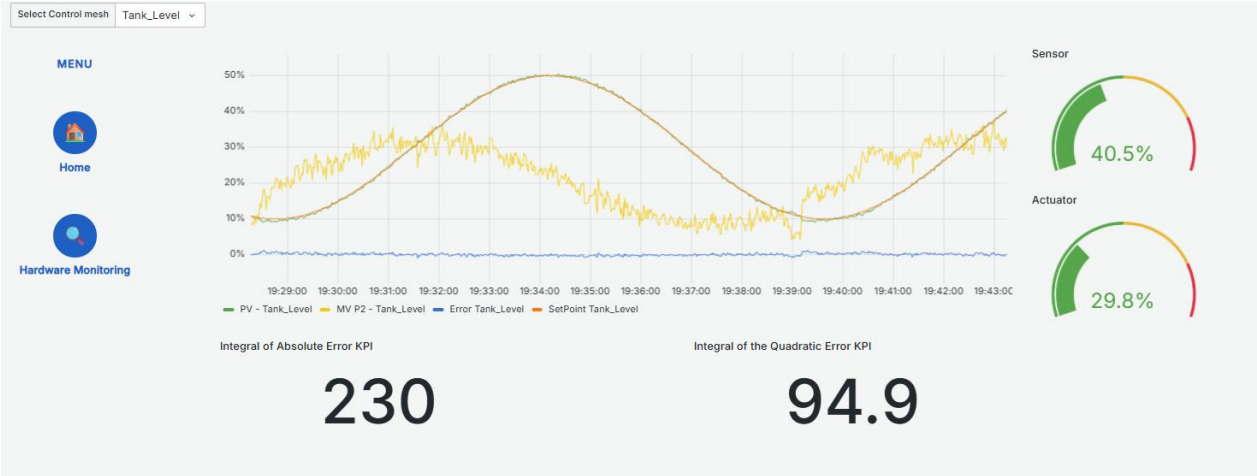
À direita do gráfico principal são apresentados dois indicadores do tipo *Gauge*, responsáveis por exibir os valores instantâneos das principais variáveis do processo. O primeiro indicador apresenta o valor atual da variável de processo (sensor), enquanto o segundo exibe o valor da variável manipulada (atuador), permitindo ao operador acompanhar rapidamente o estado atual da malha de controle.

Na parte inferior do *dashboard* são apresentados dois indicadores destinados à avaliação do desempenho do controlador, correspondentes à Integral do Erro Absoluto (IAE) e à Integral do Erro Quadrático (ISE). Essas métricas permitem quantificar a qualidade do controle executado, sendo amplamente utilizadas para comparar o desempenho de diferentes estratégias de controle e identificar possíveis degradações na resposta do sistema. As Equações (1) e (2) apresentam a formulação matemática utilizada para o cálculo desses índices.

$$IAE = \int_0^T |E(t)|\, dt \tag{1}$$

$$ISE = \int_0^T E^2(t)\, dt \tag{2}$$

Figura 24 – Dashboard control monitoring



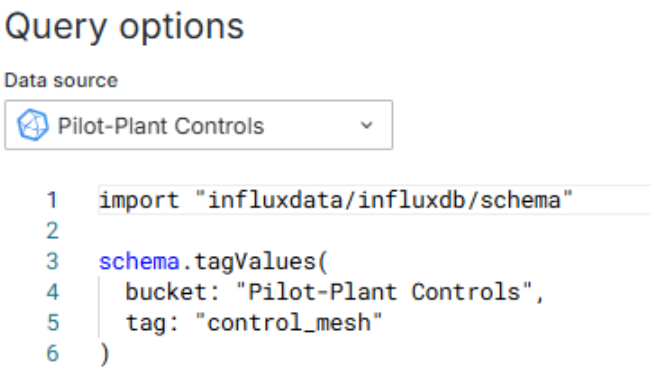
Fonte: Elaborado pelo autor

Para o monitoramento da planta piloto existem quatro malhas de controle. A fim de evitar a criação de um painel de controle específico para cada malha, foi adotada a mesma estratégia empregada no *Hardware Monitoring*, permitindo que um único painel seja utilizado para monitorar diferentes malhas de controle.

Na interface do *dashboard*, essa funcionalidade é disponibilizada por meio de um seletor localizado no canto superior esquerdo da tela, no qual o usuário pode escolher a malha de controle que deseja monitorar. Embora o princípio de funcionamento seja semelhante ao utilizado para a seleção do dispositivo (*host*), sua implementação apresenta algumas particularidades.

As variáveis de controle, como o SP e o Erro, são provenientes do mesmo microsserviço de controle, por exemplo lic125_pid4. No InfluxDB, esses dados são armazenados juntamente com a etiqueta *control_mesh,* responsável por identificar a qual malha de controle cada registro pertence. Dessa forma, foi criada no Grafana uma variável de consulta que recupera automaticamente os valores dessa *tag*, de maneira análoga à variável host apresentada anteriormente.

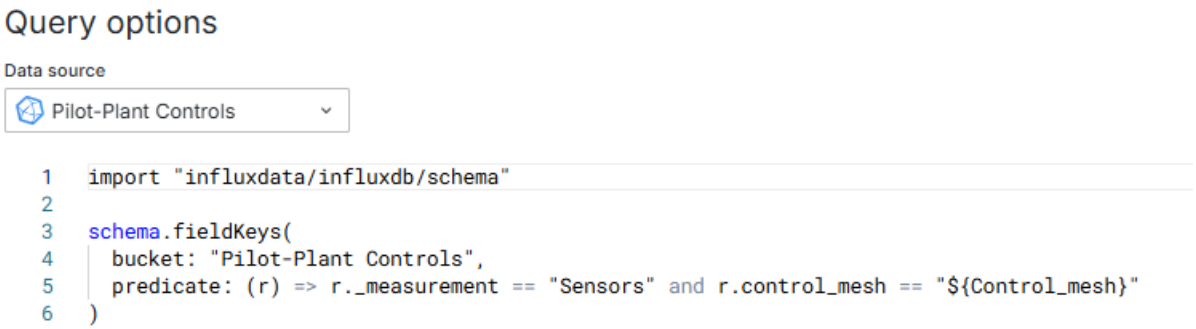
Figura 25 – Variável Grafana control mesh



Fonte: Elaborado pelo autor

A Figura 25 apresenta a configuração da variável *control_mesh*, utilizada para selecionar a malha de controle desejada. Entretanto, as variáveis correspondentes aos sensores não possuem a *tag*, impossibilitando sua associação direta à malha de controle selecionada. Para solucionar essa limitação, foi criada uma segunda variável no Grafana, responsável por associar dinamicamente os sensores às respectivas malhas de controle. Essa associação permite que, ao selecionar uma determinada malha, apenas os sensores correspondentes sejam disponibilizados e utilizados nas consultas do *dashboard*.

Figura 26 – Variável Grafana sensor



Fonte: Elaborado pelo autor

Conforme ilustrado na Figura 26, a variável Sensor consulta os campos disponíveis no banco de dados e os filtra de acordo com a malha de controle selecionada na variável *control_mesh*, estabelecendo uma relação entre ambas. Dessa forma, apenas os sensores pertencentes à malha escolhida ficam disponíveis para utilização nas consultas do *dashboard*.

Para os atuadores, a estratégia adotada foi diferente. Como alguns atuadores são compartilhados entre mais de uma malha de controle a associação não pode ser realizada apenas por meio de uma variável. Nesse caso, foi implementada uma estrutura condicional (*if*) nas consultas em Flux, responsável por selecionar automaticamente o atuador correspondente de acordo com a malha de controle escolhida pelo usuário. Dessa forma, para determinadas malhas é utilizada a bomba 1, enquanto para outras é utilizada a bomba 2, garantindo que os dados apresentados sejam compatíveis com a configuração da planta piloto.

4.7 Integração com a arquitetura PnPr

A etapa final do desenvolvimento consistiu na integração da aplicação à arquitetura PnPr. Essa integração foi realizada por meio da incorporação de duas novas abas ao painel de controle do *Produce*. A integração foi implementada no Node-RED *Dashboard*, utilizando o mecanismo de navegação por links. Dessa forma, as abas adicionadas redirecionam o usuário para os endereços IP em que estão hospedados o Grafana e o InfluxDB, permitindo o acesso às funcionalidades de monitoramento e ao historiador de dados diretamente a partir da interface principal da arquitetura PnPr.

A distribuição dos *dashboards* também foi adaptada para serem utilizáveis por meio do display disponível na planta piloto, conforme ilustrado na Figura 28. Essa adaptação elimina a necessidade de um computador dedicado para operação e contribui para a implementação de uma solução alinhada aos princípios da computação de borda.

Figura 27 – Display planta piloto Produce



Fonte: Elaborado pelo autor

5 RESULTADOS E DISCUSSÕES

Este capítulo apresenta os resultados obtidos a partir da implementação da camada de monitoramento proposta neste trabalho. Foram avaliados o desempenho das consultas realizadas pelos *dashboards* desenvolvidos no Grafana, bem como a capacidade da arquitetura em disponibilizar informações históricas das variáveis de processo e das métricas de hardware. Para garantir a comparação entre os diferentes painéis, os tempos médios apresentados nas Tabelas 3 e 4 foram obtidos considerando um intervalo de consulta de cinco minutos, no qual todos os *dashboards* foram submetidos às mesmas condições de avaliação.

A Tabela 3 apresenta o tempo médio de execução das consultas dos painéis que compõem o *Control Monitoring*. Observa-se que todos os painéis apresentaram tempos médios inferiores a 250 ms. O gráfico principal da malha de controle apresentou o maior tempo de resposta (223,10 ms), resultado esperado devido ao maior volume de informações processadas, uma vez que esse painel consulta simultaneamente as variáveis de processo, valor de referência, variável manipulada e erro de controle. Os indicadores do tipo *Gauge* e os painéis responsáveis pelo cálculo das métricas de controle IAE e ISE apresentaram tempos inferiores, demonstrando que consultas envolvendo menor quantidade de dados demandam menor tempo de processamento. A Tabela 4 apresenta o tempo médio de execução das consultas dos painéis do *Hardware Monitoring*. Todos os gráficos apresentaram tempos inferiores a 180 ms, sendo os menores valores observados para os gráficos de utilização da CPU e da memória RAM.

Tabela 3 – Tempo médio de execução das consultas dos painéis do Control Monitoring

Requisição	Tempo (ms)
Consulta da malha de controle	223.10
Consulta da variável de processo	128.40
Consulta da variável manipulada	147.60
Consulta Indicador da Integral do Erro Absoluto	159.90
Consulta Indicador da Integral do Erro Quadrático	217.40

Fonte: Elaborado pelo autor

Tabela 4 – Tempo médio de execução das consultas dos painéis do Hardware Monitoring

Requisição	Tempo (ms)
Consulta de utilização da CPU	94.20
Consulta de utilização da memória RAM	95.00
Consulta de temperatura da CPU	139.10
Consulta de utilização do disco	175.60

Fonte: Elaborado pelo autor

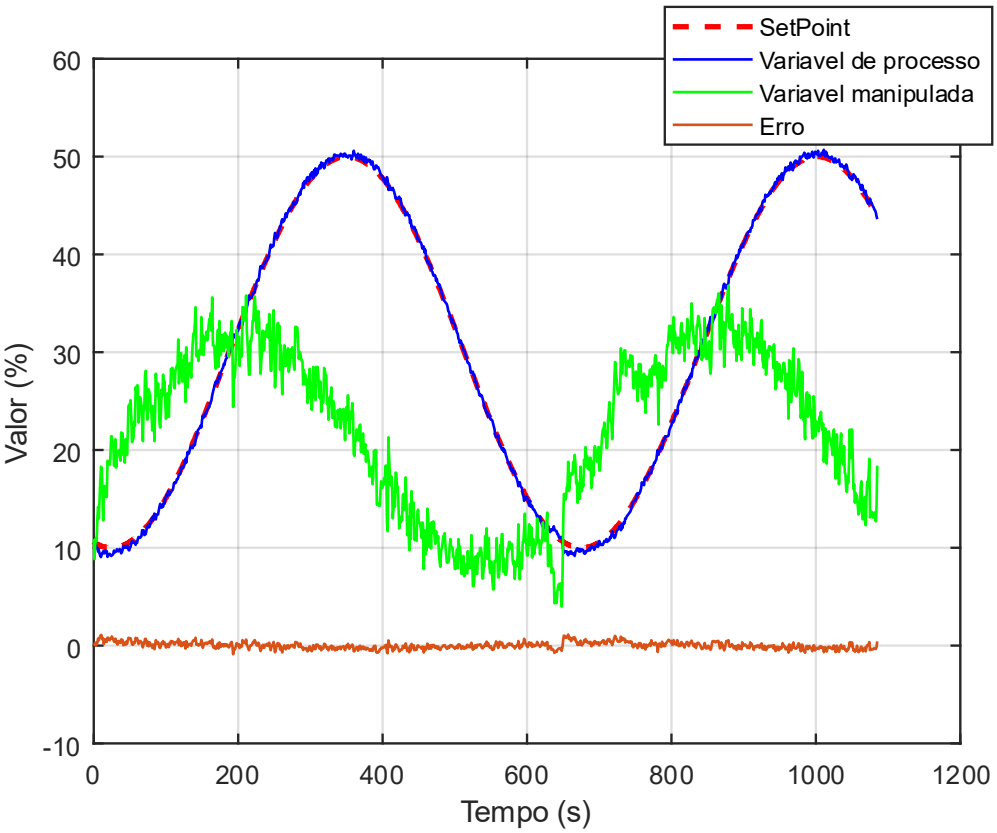
Comparando os resultados apresentados nas Tabelas 3 e 4, observa-se que o *Hardware Monitoring* apresentou tempos médios de consulta inferiores aos obtidos no *Control Monitoring*. Esse comportamento é explicado pela frequência de aquisição dos dados. Enquanto as variáveis de processo são armazenadas no InfluxDB a cada 1 segundo pelo Node-RED, as métricas de hardware são registradas pelo Telegraf em intervalos de 10 segundos. Dessa forma, considerando que as medições foram realizadas para um intervalo de consulta de cinco minutos, o *Hardware Monitoring* necessita recuperar uma quantidade significativamente menor de pontos armazenados no banco de dados, reduzindo o tempo necessário para execução das consultas.

Outro aspecto observado durante o desenvolvimento foi a facilidade de implantação da infraestrutura proposta. A utilização de contêineres docker permitiu que todos os serviços da aplicação fossem distribuídos de maneira padronizada, simplificando o processo de instalação, atualização e manutenção da arquitetura. Da mesma forma, a configuração do agente Telegraf mostrou-se facilmente replicável entre as Raspberry Pi da planta piloto, sendo necessário apenas ajustar o identificador do dispositivo monitorado. Essa característica demonstra a escalabilidade da solução desenvolvida, permitindo sua expansão para um número maior de dispositivos com reduzido esforço de configuração.

Além da visualização em tempo real, o Grafana disponibiliza mecanismos para exportação dos resultados das consultas realizadas no banco de dados. Essa funcionalidade possibilita a utilização dos dados históricos em ferramentas externas de análise, como o MATLAB, permitindo a realização de estudos complementares, desenvolvimento de algoritmos e processamento estatístico das séries temporais coletadas.

A Figura 29 apresenta um exemplo dessa utilização, no qual os dados exportados do Grafana foram empregados para construção de um gráfico de controle da planta piloto.

Figura 28 – Gráfico de controle obtido no Grafana



Fonte: Elaborado pelo autor

6 CONCLUSÃO

Este trabalho teve como objetivo desenvolver e integrar uma camada de monitoramento distribuído à arquitetura *Plug-and-Produce* baseada em microsserviços da planta piloto de Indústria 4.0. A solução implementada permitiu a aquisição, o armazenamento e a visualização histórica tanto das variáveis de processo quanto das métricas de hardware da infraestrutura, suprindo uma limitação existente na arquitetura originalmente proposta e atendendo aos objetivos estabelecidos.

Os resultados obtidos demonstraram a viabilidade da integração entre os componentes desenvolvidos e a arquitetura PnPr. A utilização conjunta do Node-RED, Telegraf, InfluxDB e Grafana possibilitou a construção de uma infraestrutura capaz de consolidar, em um único ambiente, informações provenientes dos microsserviços e dos dispositivos computacionais distribuídos. Além disso, a adoção da comunicação direta entre o Node-RED e os microsserviços por meio da biblioteca Moleculer e do *transport* NATS reduziu a dependência de componentes intermediários, tornando o fluxo de aquisição de dados mais simples e eficiente.

A infraestrutura desenvolvida amplia significativamente a capacidade de supervisão da arquitetura PnPr, permitindo não apenas acompanhar o comportamento das variáveis de processo, mas também monitorar continuamente o estado operacional dos dispositivos responsáveis pela execução da aplicação distribuída. Os experimentos realizados demonstraram que a solução apresenta desempenho compatível com aplicações de monitoramento em tempo real, com tempos de consulta inferiores a 250 ms, além de fornecer recursos para análise histórica, rastreabilidade dos dados e acompanhamento da utilização de CPU, memória, armazenamento e temperatura dos dispositivos.

Outro aspecto relevante é a escalabilidade da solução proposta. A utilização de contêineres docker, associada à configuração modular do agente Telegraf, permite incorporar novos dispositivos à infraestrutura de monitoramento com poucas alterações na arquitetura existente, favorecendo sua expansão para aplicações distribuídas de maior porte.

Como trabalhos futuros, propõe-se a implementação de mecanismos automáticos de geração de alertas baseados em limiares operacionais, a integração de técnicas de detecção de falhas e manutenção preditiva, bem como a utilização dos dados históricos armazenados para o desenvolvimento de modelos de análise e apoio à tomada de decisão em ambientes alinhados aos princípios da Indústria 4.0.

REFERÊNCIAS

CARLSSON, Oscar et al. Migration of industrial process control systems to service-oriented architectures. International Journal of Computer Integrated Manufacturing, v. 31, n. 2, p. 175-198, 2018.

CHIABURU, Mihaela. The role of time series databases in environmental research. 2026.

DOMINGUES, Felipe de Oliveira. Historiador de processos como um microsserviço: uma abordagem para a Indústria 4.0. 2021.

GRAFANA LABS. Grafana: the open and composable observability and data visualization platform. Disponível em: https://grafana.com/. Acesso em: 23 maio 2026.

GUEDES, Kaio Yukio Gonçalves Vieira. ANÁLISE DE SÉRIES TEMPORAIS: Conceitos e aplicações. 2025.

INFLUXDATA. InfluxDB: time series database. Disponível em: https://www.influxdata.com/. Acesso em: 23 maio 2026.

INFLUXDATA. What is a Data Historian and Is It Still Relevant in 2025? Disponível em: https://www.influxdata.com/glossary/data-historian/. Acesso em: 2 jul. 2026.

KHABBAZI, Mahmood Reza et al. Plug and Produce—a review and future trend. The International Journal of Advanced Manufacturing Technology, v. 134, n. 9, p. 3991-4014, 2024.

LARRINAGA, Felix et al. Node-red workflow manager for edge service orchestration. In: NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium. IEEE, 2022. p. 1-6.

DE LIMA, Marcelo Costa; LICHTNOW, Daniel. Avaliando a Performance de SGBDs na Inserção e Consulta de Dados de Séries Temporais. In: Escola Regional de Banco de Dados (ERBD). SBC, 2024. p. 170-173.

LYDON, Bill. Historiadores industriais fornecem dados para eficiência da operação. Aquarius Software, 2017. Disponível em: https://www.aquarius.com.br/historiadores-industriais-fornecem-dados-para-eficiencia-da-operacao/. Acesso em: 28 jun. 2026.

MOLECULER. Moleculer: Progressive microservices framework for Node.js. Disponível em: https://moleculer.services/. Acesso em: 23 maio 2026.

NODE-RED. Flow-based programming for the Internet of Things. Disponível em: https://nodered.org/. Acesso em: 23 maio 2026.

PIRES, Luis Fernando Dos Santos. BANCO DE DADOS PARA IOT. Revista de Ubiquidade, v. 8, n. 2, p. 23-38, 2025.

PONTAROLLI, Ricardo P. et al. Planta piloto 4.0: Uma abordagem para a automação e controle de processos orientada a serviços. In: Congresso Brasileiro de Automática-CBA. 2020.

PONTAROLLI, Ricardo Pasquati et al. Microservice-oriented architecture for industry 4.0. Eng, v. 4, n. 2, p. 1179-1197, 2023.

PONTAROLLI, Ricardo P. et al. Proposal of a microservice plug-and-produce architecture for industry 4.0. In: 2024 IEEE International Workshop on Metrology for Industry 4.0 & IoT (MetroInd4. 0 & IoT). IEEE, 2024. p. 198-203.

PONTAROLLI, Ricardo Pasquati. Arquitetura plug-and-produce orientada a microsserviços para automação na indústria 4.0. 2025.

RASPBERRY PI. Raspberry Pi. Disponível em: https://www.raspberrypi.com/. Acesso em: 27 jun. 2026.

SANTANA, João Vitor Rocha. Aplicativo para monitoramento e visualização de processos na indústria 4.0. 2022.

SEQUENT MICROSYSTEMS. Building Automation Card for Raspberry Pi. Disponível em: https://sequentmicrosystems.com/index.php?route=product/product&path=20&product_id=52. Acesso em: 23 maio 2026.

SENNA, Pedro P. et al. Prioritizing barriers for the adoption of Industry 4.0 technologies. Computers & Industrial Engineering, v. 171, p. 108428, 2022.

SILVA, Gabriel Sales Mendes da. Digitalização na indústria 4.0-análise de digitalização de fábricas: implementação de sistemas de monitoramento, coleta de dados e automação. 2023.