

# Validação de uma Infraestrutura Computacional Elástica com K3s em Dispositivos de Borda para Monitoramento Industrial

Luis Felipe da Silva Batista, Edie C. Santana

Instituto de Computação – Universidade Federal de Mato Grosso (UFMT)  
Cuiabá – MT – Brasil

Secretaria de Ciência, Tecnologia e Inovação de Mato Grosso  
Cuiabá – MT – Brasil

luisfelipe66@gmail.com, edie.santana@secitec.mt.gov.br

**Abstract.** *This paper proposes and validates the implementation of a scalable computational infrastructure for real-time monitoring of electric boilers, using Kubernetes (K3s) clusters on low-cost devices (Raspberry Pi). The central problem addresses how to provide a monitoring solution that is both energy-efficient and capable of dynamically adapting to workload variations. The main objective is to deploy and validate a solution that optimizes the use of computational resources through container orchestration and horizontal autoscaling. The methodology includes defining a use case, describing the hardware and software architecture, and conducting a case study to analyze the effectiveness of the solution's horizontal scalability. The results demonstrate the feasibility and efficiency of the approach, confirming that the cluster responded adequately to increases and decreases in load, adjusting resources autonomously.*

**Resumo.** *Este trabalho propõe e valida a implantação de uma infraestrutura computacional escalável para o monitoramento em tempo real de caldeiras elétricas, utilizando clusters Kubernetes (K3s) em dispositivos de baixo custo (Raspberry Pi). A problemática central aborda como prover uma solução de monitoramento que seja ao mesmo tempo energeticamente eficiente e capaz de se adaptar dinamicamente a variações na carga de trabalho. O objetivo geral é implantar e validar uma solução que otimize o uso de recursos computacionais através da orquestração de contêineres e do autoescalonamento horizontal. A metodologia inclui a definição de um cenário de uso, a descrição da arquitetura de hardware e software, e a realização de um estudo de caso para analisar a eficácia da escalabilidade horizontal da solução. Os resultados demonstram a viabilidade e a eficiência da abordagem, confirmando que o cluster respondeu adequadamente aos aumentos e diminuições de carga, ajustando os recursos de forma autônoma.*

## 1. Introdução

A demanda por sistemas de monitoramento industrial que aliem alta disponibilidade, baixo custo e eficiência energética é crescente. Tradicionalmente, o monitoramento de ativos como caldeiras elétricas depende de servidores locais robustos, que operam com capacidade provisionada para picos de carga. Embora confiáveis, essas arquiteturas apresentam desafios significativos, como o alto custo de aquisição e manutenção, o consumo

elevado de energia e uma complexidade inerente para escalar [2]. A necessidade de adicionar mais pontos de monitoramento exige um planejamento de capacidade que nem sempre é ágil ou economicamente viável.

É neste ponto que a tecnologia de orquestração de contêineres, liderada pelo Kubernetes [5], oferece uma mudança de paradigma. Ao abstrair a aplicação da infraestrutura, o Kubernetes permite a criação de sistemas distribuídos, resilientes e, fundamentalmente, elásticos. Esta capacidade de adaptação abre portas para o uso de hardware de baixo custo, como o Raspberry Pi, em cenários industriais antes dominados por servidores de alto desempenho, alinhando-se aos princípios da Computação de Borda (Edge Computing).

Diante desse cenário, a problemática investigada é: como prover uma infraestrutura computacional escalável, de baixo custo e energeticamente eficiente para o monitoramento em tempo real de caldeiras elétricas? A lacuna existente está na dificuldade de alocar e desalocar dinamicamente os recursos de processamento para atender às variações de demanda sem comprometer a eficiência operacional.

Com base nisso, o objetivo geral deste trabalho consiste em implantar e validar uma solução computacional escalável utilizando Kubernetes, sobre dispositivos de baixo custo (Raspberry Pi), para o monitoramento da temperatura de caldeiras elétricas, buscando aliar economia de energia à alocação dinâmica e eficiente de recursos computacionais.

Para alcançar o objetivo maior, os seguintes passos específicos foram definidos:

1. Investigar as soluções de monitoramento de caldeiras, compreendendo as arquiteturas tradicionais e seus desafios.
2. Analisar as tecnologias de implantação de Kubernetes em dispositivos de baixo custo, com foco no Raspberry Pi.
3. Implantar uma solução computacional funcional com Kubernetes (K3s) para a coleta e processamento de dados de temperatura.
4. Validar a escalabilidade horizontal do cluster em um estudo de caso que simula a variação no número de caldeiras monitoradas.

## **2. Fundamentação Teórica**

### **2.1. Virtualização e Contêineres**

A virtualização tradicional, baseada em hipervisores, permite que um servidor físico hospede múltiplas máquinas virtuais (VMs), cada uma com seu próprio sistema operacional. Em contraste, os contêineres oferecem virtualização a nível de sistema operacional, onde múltiplos contêineres compartilham o mesmo kernel do host, tornando-os significativamente mais leves, rápidos e portáteis [9].

### **2.2. Orquestração com Kubernetes e Escalabilidade Horizontal**

Gerenciar o ciclo de vida de centenas de contêineres manualmente é impraticável. O Kubernetes surge como a solução para este desafio, orquestrando contêineres em um cluster de máquinas. Sua função crucial é a escalabilidade. A escalabilidade horizontal (ou scale-out) consiste em aumentar ou diminuir o número de réplicas (pods) de uma aplicação em resposta a métricas predefinidas, como o uso de CPU, sendo o cerne da elasticidade do sistema [10].

### 2.3. Kubernetes na Computação de Borda

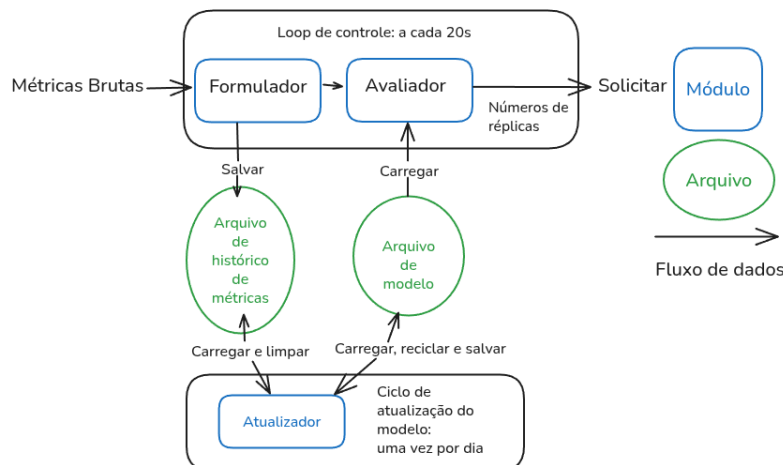
A aplicação de Kubernetes em dispositivos de recursos limitados, como o Raspberry Pi, tornou-se um campo de pesquisa ativo. Distribuições leves como o K3s foram desenvolvidas para minimizar o consumo de recursos, tornando viável a criação de clusters de borda para aplicações de IoT e monitoramento [6]. Estudos comparativos demonstram que, para cargas de trabalho de baixa a média intensidade, clusters de Raspberry Pi gerenciados por K3s apresentam um desempenho satisfatório com um custo energético e financeiro drasticamente reduzido em comparação com soluções em nuvem ou servidores tradicionais [3]. O autoescalonamento nestes ambientes é chave para garantir a responsividade sem esgotar os recursos limitados dos dispositivos [1].

## 3. Proposta da Solução e Metodologia

Esta seção apresenta a solução proposta, detalhando a arquitetura de monitoramento construída sobre um cluster de Raspberry Pi com K3s. Descreve também os componentes de hardware e software utilizados e a metodologia de validação focada no autoescalonamento horizontal.

### 3.1. Cenário de Uso

O cenário simula um ambiente de monitoramento industrial onde um cluster de dispositivos Raspberry Pi deve escalar dinamicamente os processos (pods) de análise de dados, conforme a demanda de sensores de temperatura. A arquitetura da solução (Figura 1) utiliza um ciclo de controle onde um Avaliador, com base em um modelo preditivo, determina o número de réplicas necessárias, enquanto um Atualizador retreina o modelo periodicamente para garantir a alocação proativa de recursos.



Fonte: Adaptado de Ju, Singh e Toor (2021).

**Figura 1. Arquitetura da solução proposta, ilustrando o fluxo de dados e o mecanismo de autoescalonamento.**

### 3.2. Ambiente Experimental

O ambiente experimental foi construído com hardware de baixo custo e software open-source. O cluster foi formado por três nós Raspberry Pi 4 (4GB RAM, CPU Quad-core Cortex-A72) [8] rodando Raspberry Pi OS (64-bit), com um nó atuando como *master*

K3s e dois como *workers*. Sensores de temperatura DS18B20 foram conectados às portas GPIO para a coleta de dados. A arquitetura de software utilizou K3s [4] como orquestrador de contêineres, RabbitMQ [7] como message broker para desacoplar os dados, e uma aplicação em Python containerizada (Docker) para processar as informações da fila.

### 3.3. Estudo de Caso: Análise de Escalabilidade

Para validar a elasticidade da solução, a aplicação de monitoramento foi implantada no cluster K3s com um limite de ‘500m’ (meio core de CPU) e configurada com um Horizontal Pod Autoscaler (HPA). O gatilho para o escalonamento foi o uso de CPU, definido para ser ativado quando a utilização média atingisse 70% da capacidade alocada. O HPA foi configurado para manter no mínimo 1 pod e no máximo 4 pods.

## 4. Resultados e Análise

O estudo de caso foi dividido em dois cenários para observar o comportamento de *scale-out* (escalar para cima) e *scale-in* (escalar para baixo).

### 4.1. Cenário 1: Escalabilidade Horizontal Incremental (Scale-Out)

O teste iniciou com um simulador enviando dados equivalentes a **uma caldeira** para a fila do RabbitMQ. O K3s manteve uma única réplica (pod) da aplicação, cujo consumo de CPU estabilizou em aproximadamente **35%** do recurso alocado (cerca de 175m de CPU).

Em seguida, a carga foi aumentada para simular **três caldeiras** ativas. O consumo de CPU do único pod saltou para aproximadamente **85%** (cerca de 425m), ultrapassando o limite de 70% definido no HPA.

**Análise:** Conforme esperado, após o período de observação padrão do Kubernetes, o HPA iniciou o processo de escalonamento. Em aproximadamente 40 segundos, um segundo pod foi provisionado e entrou em estado *Running*. A carga de trabalho foi então distribuída pelo balanceador de carga do Kubernetes entre os dois pods, e o consumo de CPU em cada um deles estabilizou em torno de **43%**, demonstrando a eficácia do *scale-out* para lidar com o aumento da demanda.

### 4.2. Cenário 2: Escalabilidade Horizontal Decremental (Scale-In)

Partindo do estado anterior (dois pods ativos), a carga de trabalho foi reduzida de volta para o equivalente a **uma caldeira**. A utilização de CPU em ambos os pods caiu para aproximadamente **18%** cada.

**Análise:** O HPA mantém um período de estabilização para evitar o *flapping* (escalar para cima e para baixo repetidamente). Após a janela de estabilização padrão do K3s (padrão de 5 minutos) com a carga consistentemente baixa, o HPA identificou o pod como ocioso e iniciou o processo de *scale-in*. Um dos pods foi terminado, e o sistema retornou ao estado inicial com um único pod consumindo aproximadamente 35% de CPU. Este teste validou a capacidade do sistema de economizar recursos ao reduzir a infraestrutura em resposta à diminuição da demanda.

### 4.3. Discussão dos Resultados

Os resultados práticos do estudo de caso confirmam a tese central deste trabalho: é viável e vantajoso utilizar uma arquitetura baseada em K3s em dispositivos de baixo custo para

monitoramento industrial. A capacidade do HPA de provisionar e remover pods de forma autônoma, baseada em métricas de uso real, valida a elasticidade da solução. A arquitetura proposta não é apenas uma alternativa mais econômica em termos de hardware, mas também mais eficiente e sustentável do ponto de vista energético, pois o consumo de recursos computacionais acompanha dinamicamente a carga de trabalho.

## 5. Conclusão

Este trabalho demonstrou a concepção, implementação e, crucialmente, a validação de uma solução escalável e de baixo custo para monitoramento industrial, utilizando Kubernetes (K3s) em um cluster de Raspberry Pi. A arquitetura proposta provou ser capaz de se adaptar dinamicamente às flutuações de carga, atendendo aos requisitos de eficiência e otimização de recursos por meio da escalabilidade horizontal.

A validação por meio do estudo de caso confirmou que a abordagem é tecnicamente robusta e responsiva, posicionando-se como uma alternativa moderna e vantajosa em relação às infraestruturas centralizadas tradicionais para aplicações de Computação de Borda. O trabalho abre caminho para futuras investigações, incluindo a análise de outras métricas de escalonamento (como métricas customizadas via Prometheus) e a integração de mecanismos de tolerância a falhas no cluster.

## Referências

- [1] Gama, K. et al. (2021). A Survey on Autonomic Provisioning of Fog Computing Services. *ACM Computing Surveys*, 54(5), 1-38.
- [2] Ghosh, A. et al. (2021). A Review on Industrial IoT: Applications, Technologies, and Challenges. *IEEE Access*, 9, 45939-45973.
- [3] Ismail, B. I.; Khan, S. (2021). Performance Evaluation of Lightweight Kubernetes Distributions for Edge Computing. In: *2021 IEEE International Conference on Edge Computing (EDGE)*, pp. 1-8.
- [4] K3S. (2025). *Lightweight Kubernetes*. Disponível em: <https://k3s.io/>. Acesso em: 10 out. 2025.
- [5] Kubernetes. (2025). *Official Documentation*. Disponível em: <https://kubernetes.io/docs/home/>. Acesso em: 10 out. 2025.
- [6] Mora, H. et al. (2022). Performance Analysis of K3s, MicroK8s, and Kubeadm for Deploying Kubernetes Clusters on the Edge. *Sensors*, 22(19), 7354.
- [7] RabbitMQ. (2025). *Messaging that just works*. Disponível em: <https://www.rabbitmq.com/>. Acesso em: 10 out. 2025.
- [8] Raspberry Pi Foundation. (2025). *Raspberry Pi Documentation*. Disponível em: <https://www.raspberrypi.com/documentation/>. Acesso em: 10 out. 2025.
- [9] Merkel, D. (2014). Docker: lightweight linux containers for consistent development and deployment. *Linux Journal*, 2014(239), 2.
- [10] Nguyen, T.-T. et al. (2020). Horizontal Pod Autoscaling in Kubernetes for Elastic Container Orchestration. *Sensors*, 20(16), 4621.