UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Modelos arquiteturais para sintetização de fluxo de dados para aplicações de pequena, média e larga escala no contexto de Internet das Coisas (IoT)

Cairo Mateus Neves Ribeiro

Dissertação de Mestrado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP
SÃO CARLOS

Cairo Mateus Neves Ribeiro

Modelos arquiteturais para sintetização de fluxo de dados para aplicações de pequena, média e larga escala no contexto de Internet das Coisas (IoT)

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. *EXEMPLAR DE DEFESA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Júlio Cezar Estrella

USP – São Carlos
Outubro de 2023

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

R484m

Ribeiro, Cairo Mateus Neves
Modelos arquiteturais para sintetização de fluxo de dados para aplicações de pequena, média e larga escala no contexto de Internet das Coisas (IoT) / Cairo Mateus Neves Ribeiro; orientador Julio Cezar Estrella. -- São Carlos, 2023.
93 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2023.

1. Internet das Coisas. 2. Arquiteturas de Internet das Coisas. 3. Fluxos de Dados Fluxos de Dados. I. Cezar Estrella, Julio, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Cairo Mateus Neves Ribeiro

Architectural models for data flow synthesis for small, medium and large-scale applications in the context of the Internet of Things (IoT)

Master dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP, in partial fulfillment of the requirements for the degree of the Master Program in Computer Science and Computational Mathematics. *EXAMINATION BOARD PRESENTATION COPY*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Júlio Cezar Estrella

USP – São Carlos
October 2023

AGRADECIMENTOS

Presto meus agradecimentos à toda equipe do Departamento de Sistemas de Computação (SSC) do ICMC, pelo acolhimento e auxilio nesta etapa do mestrado. Agradeço também, aos parceiros de trabalho e alunos de graduação que tive o prazer de dividir meu tempo e um pouco do conhecimento adquirido nesse ambiente de ensino. Entre eles, Beatriz Fialho, Bruno Mazzoti e Júlio Bueno, não se limitando a esses parceiros e amigos.

Agradeço ao meu orientador, Júlio Cezar Estrella pela paciência e sugestões de como conduzir esse trabalho da melhor forma possível. Por fim, agradeço a minha família pelo cuidado e atenção nessa fase de minha vida.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001

“Nem tudo o que reluz é ouro,
Nem todos os que vagueiam estão perdidos;
(J. R. R. Tolkien)”

RESUMO

RIBEIRO, C. N. **Modelos arquiteturais para sintetização de fluxo de dados para aplicações de pequena, média e larga escala no contexto de Internet das Coisas (IoT)**. 2023. 93 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

O número de dispositivos inteligentes conectados a Internet cresce a cada ano e estima-se que em 2025 haja 28 bilhões deles em operação. A criação de arquiteturas que consigam gerenciar esse imenso fluxo de dados se tornam cada vez mais necessárias. O objetivo deste trabalho é realizar um estudo de como a escala da aplicação impacta no projeto de arquitetura para aplicações de Internet das Coisas. Para atingir esse objetivo foi realizado inicialmente um estudo de diversas arquiteturas apresentadas em análises bibliográficas e apresentamos as principais diferenças entre arquiteturas de pequeno, médio e grande porte. Neste contexto, a proposta é apresentar arquiteturas genéricas para pequeno, médio e grande porte de modo que seja realizado o desenvolvimento e a implementação da arquitetura de médio porte determinando/registrando o fluxo de dados e quais os pontos de atenção ao desenvolver aplicações deste contexto. Análises no protótipo de médio porte são realizadas e os resultados para essas análises são apresentadas com o objetivo de demonstrar suas capacidades de escritas e leituras. Estas análises foram realizadas em um controle distribuído de sistema de ares-condicionados que é descrito e analisado no decorrer da dissertação. Por fim, finalizamos descrevendo como este projeto pode ser estendido e continuado.

Palavras-chave: Internet das Coisas, Arquiteturas de Internet das Coisas, Fluxos de Dados.

ABSTRACT

RIBEIRO, C. N. **Architectural models for data flow synthesis for small, medium and large-scale applications in the context of the Internet of Things (IoT)**. 2023. 93 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

The number of smart devices connected to the Internet grows every year, and it is estimated that by 2025 there will be 28 billion of them in operation. The creation of architectures that can manage this immense flow of data is becoming increasingly necessary. The objective of this work is to study how the scale of the application impacts the architecture design for Internet of Things applications. To achieve this goal, a study of various architectures presented in bibliographic analyses was initially conducted, and the main differences between small, medium, and large-scale architectures were presented. In this context, the proposal is to present generic architectures for small, medium, and large-scale applications, such that the development and implementation of the medium-scale architecture are determined/recorded, along with the data flow and the key points to consider when developing applications in this context. Analyses on the medium-scale prototype are conducted, and the results for these analyses are presented with the aim of demonstrating their writing and reading capabilities. These analyses were performed on a distributed control system for air conditioners, which is described and analyzed throughout the dissertation. Finally, we conclude by describing how this project can be extended and continued.

Keywords: Internet of Things, Internet of Things Architectures, Data Streams.

LISTA DE ILUSTRAÇÕES

Figura 1 – Protocolo MQTT 29
Figura 2 – Aplicações de pequena para grande escala em IoT adaptado de Sunhare, Chowdhary e Chattopadhyay (2020) 30
Figura 3 – Relação dos trabalhos depois da primeira seleção 38
Figura 4 – Relação dos trabalhos depois da segunda seleção 39
Figura 5 – Arquitetura proposta por Malche e Maheshwary (2017) 40
Figura 6 – Arquitetura proposta por Rathore *et al.* (2016) 43
Figura 7 – Proposta da arquitetura para pequeno porte. 46
Figura 8 – Fluxo de dados da arquitetura de pequena escala 48
Figura 9 – Proposta da arquitetura para médio porte. 49
Figura 10 – Fluxo de dados da arquitetura de média escala 51
Figura 11 – Proposta da arquitetura para grande porte. 52
Figura 12 – Fluxo de dados da arquitetura de grande escala 53
Figura 13 – Arquitetura de Médio Porte 56
Figura 14 – Ar condicionado ajustado no ambiente físico para a arquitetura 57
Figura 15 – Dispositivos utilizados no ambiente do laboratório 58
Figura 16 – Tela de listagem de microcontroladores 60
Figura 17 – Arquitetura do Kafka 62
Figura 18 – Descrição visual das *Bridges* 63
Figura 19 – Ações de ponta-a-ponta do modelo arquitetura proposto 65
Figura 20 – Distribuição do tempo de resposta HTTP para o 1º Experimento 69
Figura 21 – Distribuição do tempo de resposta HTTP para o 2º Experimento 70
Figura 22 – Comparação do tempo de resposta entre do 1º Experimento com o 2º Experimento 70
Figura 23 – Adaptado da arquitetura proposta por Malche e Maheshwary (2017) 83
Figura 24 – Adaptado da arquitetura proposta por Iqbal *et al.* (2018a) 84
Figura 25 – Adaptado da arquitetura proposta por Iqbal *et al.* (2018b) 84
Figura 26 – Adaptado da arquitetura proposta por Petnik e Vanus (2018) 85
Figura 27 – Adaptado da arquitetura proposta por Amadeo *et al.* (2017) 85
Figura 28 – Adaptado da arquitetura proposta por Rahmani *et al.* (2018) 87
Figura 29 – Adaptado da arquitetura proposta por Catarinucci *et al.* (2015) 88
Figura 30 – Adaptado da arquitetura proposta por Wen *et al.* (2018) 88
Figura 31 – Adaptado da arquitetura proposta por Bharadwaj, Rego e Chowdhury (2016) 89

Figura 32 – Adaptado da arquitetura proposta por Khoi *et al.* (2015) 89
Figura 33 – Adaptado da arquitetura proposta por Rathore *et al.* (2016) 91
Figura 34 – Adaptado da arquitetura proposta por Montori, Bedogni e Bononi (2018) . . 92
Figura 35 – Adaptado da arquitetura proposta por Viswanath *et al.* (2016) 92
Figura 36 – Adaptado da arquitetura proposta por Park *et al.* (2019) 93

LISTA DE TABELAS

Tabela 1 – Configurações dos Hosts Físicos 59
Tabela 2 – Distribuição de Aplicações para as VMs 59
Tabela 3 – Entradas para o 1º Experimento 66
Tabela 4 – Entradas para o 2º Experimento 67
Tabela 5 – Resultados do 1º Experimento 68
Tabela 6 – Resultados do 2º Experimento 68
Tabela 7 – MQTT - Taxa de requisições aceitas 69
Tabela 8 – Tabela de artigos de pequena escala 81
Tabela 9 – Tabela de artigos de média escala 82
Tabela 10 – Tabela de artigos de grande escala 82

LISTA DE ABREVIATURAS E SIGLAS

BibFluxLab	Biblioteca de Controle de Fluxo de Laboratório
BIM	Modelagem da Informação da Construção
CIB ISCBE	CIB International Conference On Smart Built Environment
CIoT	Internet das Coisas Cognitiva
HTTP	Protocolo de Transferência de Hipertexto
IAU	Instituto de Arquitetura e Urbanismo
IoT	Internet das Coisas
IP	Protocolo de Rede
LaSDPC	Laboratório de Sistemas Distribuídos e Programação Concorrente
LoRa	Longo Alcance
MQTT	Transporte de telemetria de enfileiramento de mensagens
SSC	Departamento de Sistemas de Computação
TCP	Protocolo de Controle de Transmissão
UDP	Protocolo de Datagrama de Usuário
UUID	Identificador Único Universal
VMs	Máquinas Virtuais

SUMÁRIO

1 INTRODUÇÃO 23
1.1 Contextualização 23
1.2 Motivações 23
1.3 Objetivos 24
1.3.1 Objetivos Específicos *24*
1.4 Organização do Trabalho 24

2 REFERENCIAL TEÓRICO 25
2.1 Considerações Iniciais 25
2.2 Internet das Coisas (IoT) 25
2.3 Camadas de Rede 26
2.4 Protocolos da Camada de Aplicação 27
2.4.1 HTTP *28*
2.4.2 MQTT *28*
2.5 Protocolos da Camada de Transporte 29
2.6 Aplicações de IoT 30
2.6.1 Pequena Escala *31*
2.6.1.1 Casas Inteligentes (Smart Homes) *31*
2.6.2 Média Escala *31*
2.6.2.1 Edifícios/Construções Inteligentes (Smart Buildings) *32*
2.6.2.2 Transporte Inteligente (Smart Transport) *32*
2.6.2.3 Saúde Inteligente (Smart Healthcare) *32*
2.6.3 Grande Escala *33*
2.6.3.1 Cidades Inteligentes (Smart Cities) *33*
2.7 Microcontroladores 33
2.7.1 ESP-32 . . . . . *34*
2.8 Sistemas de Mensageria 34
2.9 Considerações Finais 35

3 TRABALHOS RELACIONADOS 37
3.1 Considerações Iniciais 37
3.2 Revisão da Literatura 37
3.2.1 Arquiteturas de Pequena Escala *39*

3.2.2 *Arquiteturas de Média Escala* *41*
3.2.3 *Arquiteturas de Grande Escala* *43*
3.3 Considerações Finais 44

4 DESENVOLVIMENTO DO MODELO ARQUITETURAL PARA SINTETIZAÇÃO DE FLUXO DE DADOS 45
4.1 Considerações Iniciais 45
4.2 Mapeamento das Arquiteturas 45
4.2.1 *Pequena Escala* *45*
4.2.2 *Média Escala* *48*
4.2.3 *Larga Escala* *50*
4.3 Considerações Finais 52

5 PROTOTIPAÇÃO DO MODELO ARQUITETURAL 55
5.1 Considerações Iniciais 55
5.2 Descrição do Protótipo 55
5.3 Hardware e Dispositivos 56
5.3.1 *Biblioteca de Controle de Fluxo de Laboratório* *57*
5.3.2 *Nuvem Computacional* *58*
5.4 Front-end 60
5.5 Back-end 61
5.5.1 *Bridges* *61*
5.5.2 *Banco de Dados* *63*
5.6 Considerações Finais 64

6 RESULTADOS 65
6.1 Considerações Iniciais 65
6.2 Descrição do 1º Experimento 66
6.3 Descrição do 2º Experimento 67
6.4 Resultados 68
6.5 Considerações Finais 71

7 CONCLUSÃO 73
7.1 Trabalhos Publicados 74

8 TRABALHOS FUTUROS 75

REFERÊNCIAS 77

APÊNDICE A TABELAS COM OS TRABALHOS RELACIONADOS 81

APÊNDICE B SÍNTESE DAS ARQUITETURAS DE PEQUENO PORTE 83

APÊNDICE C SÍNTESE DAS ARQUITETURAS DE MÉDIO PORTE 87

APÊNDICE D SÍNTESE DAS ARQUITETURAS DE GRANDE PORTE 91

CAPÍTULO

1

INTRODUÇÃO

1.1 Contextualização

O número de dispositivos conectados à Internet continua a cresce de maneira continua, aliado a isso a capacidade dos dispositivos de trocar dados entre si, apresenta desafios significativos para as plataformas de software. Essas plataformas precisam ser capazes de comunicar e transmitir dados entre dispositivos que utilizam diferentes protocolos de comunicação.

Para lidar com essa complexidade, as plataformas de software são divididas em diversos componentes, e deve ter uma função bem definida dentro da arquitetura. Essa separação dos componentes simplifica a comunicação e estabelece um fluxo estruturado de compartilhamento de informações.

No entanto, é importante destacar que o fluxo de compartilhamento de informações pode se tornar sobrecarregado em pontos específicos da arquitetura. Essa sobrecarga pode resultar em lentidão nos serviços fornecidos pela plataforma e, em situações mais graves, levar a perdas significativas de dados. Para evitar esses problemas, é essencial monitorar e acompanhar o fluxo de dados entre os componentes da arquitetura da plataforma, a fim de mitigar falhas e reduzir erros.

1.2 Motivações

A Internet das Coisas (IoT) possibilitou o aumento no número de dispositivos conectados a rede mundial de computadores. Estes dispositivos possuem diferentes formatos de dados, protocolos de comunicação e tempos de resposta. Com o objetivo de conectar esses diferentes dispositivos e protocolos, foram sugeridas diversas arquiteturas como pode ser visto em Bansal e Kumar (2020).

Há uma grande variedade em dispositivos, protocolos, arquiteturas e plataformas em

Internet das Coisas nos quais, o maior objetivo é a captação de dados e a tomada de decisões com esses dados. Para monitorar a captação de dados e tomar decisões de forma mais inteligente é necessário um controle de como esse dado é gerado bem como a capacidade do dispositivo de realizar o processamento das informações. Além da forma como o dado é gerado, é importante levar em consideração como ele é transmitido, por quais camadas e diferentes softwares ele transita, além, de como esse dado é armazenado.

1.3 Objetivos

Modelagem de uma arquitetura de software genérica para aplicações de pequena, média e grande escala em IoT considerando a heterogeneidade desse tipo de ambiente. Na sequência uma instância inicial do modelo será apresentata e testada.

1.3.1 Objetivos Específicos

- Traçar fluxos de dados para as diferentes tamanhos/escalas de aplicações em Internet das Coisas;
- Avaliar aplicação em Internet das Coisas para controle de equipamento de ar-condicionado;
- Discutir como a escala da aplicação afeta diretamente no projeto de soluções em Internet das Coisas.

1.4 Organização do Trabalho

A estrutura do presente trabalho é a seguinte. O Capítulo 2 discorre sobre conceitos fundamentais, incluindo IoT, camadas de rede e protocolos das camadas de aplicação e de rede, fornecendo uma visão abrangente das arquiteturas de IoT para aplicações de diferentes escalas. O Capítulo 3 detalha a metodologia adotada para a seleção dos trabalhos relacionados a esta dissertação. Em seguida, são descritas as arquiteturas de aplicações de pequeno, médio e grande porte desenvolvidas em diversos ambientes. No Capítulo 4, são apresentadas as propostas de arquitetura de pequeno, médio e grande porte, derivadas da análise dos trabalhos relacionados. Este capítulo também inclui fluxogramas ilustrando a trajetória lógica dos dados em cada uma dessas arquiteturas. No Capítulo 5, é demonstrado um estudo de caso das arquiteturas de médio porte no Laboratório de Sistemas Distribuídos e Programação Concorrente (LaSDPC) , com a apresentação dos componentes físicos e de software necessários para sua implementação. Os resultados obtidos com o desenvolvimento deste trabalho são apresentados no Capítulo 6. O Capítulo 7 traz as considerações finais do autor sobre o trabalho, enquanto o Capítulo 8 sugere direções futuras para pesquisas envolvendo este trabalho.

CAPÍTULO

2

REFERENCIAL TEÓRICO

2.1 Considerações Iniciais

Nesta seção, abordaremos a Internet das Coisas (IoT), as camadas de aplicação de rede, os protocolos utilizados em IoT organizados por camadas de rede e uma análise das aplicações de IoT em diferentes escalas.

Vamos explorar o conceito da Internet das Coisas, que se refere à interconexão de dispositivos inteligentes e sensores que coletam e trocam dados através da rede. Discutiremos as camadas de aplicação de rede, que são os diferentes níveis de funcionalidades em uma arquitetura IoT, incluindo a camada física, de rede e de aplicação.

Além disso, iremos analisar os protocolos utilizados em IoT, que são os conjuntos de regras e formatos de comunicação que possibilitam a troca de informações entre dispositivos. Esses protocolos serão abordados em relação a cada camada de rede, destacando sua importância e características.

Vamos explorar as aplicações de IoT em diferentes escalas. Discutiremos como a IoT pode ser aplicada em nível pessoal, residencial, corporativo e até mesmo em cidades inteligentes, destacando os benefícios e desafios enfrentados em cada contexto. Com isso, teremos uma visão abrangente das possibilidades e impactos da IoT em diferentes áreas da sociedade. Por fim, vamos contextualizar microcontroladores e sistemas de mensageria.

2.2 Internet das Coisas (IoT)

A IoT é uma tecnologia que conecta uma ampla gama de dispositivos e arquiteturas de software com o objetivo de solucionar problemas em diversas áreas de aplicação. De acordo com Bansal e Kumar (2020) a Internet das coisas (IoT) é uma tecnologia em que bilhões de dispositivos interconectados, incluindo sensores, atuadores, *gateways* e controladores, que

trabalham em conjunto para fornecer eficientemente aplicações para diversos domínios.

Segundo, Gupta e Quamara (2020) aplicações de IoT se dividem em três grupos principais: coleta de informações, análise de dados e aplicações de tempo real para a tomada de decisões. As aplicações de coleta de informações têm a responsabilidade de obter dados dos dispositivos e armazenamentos locais. Já as aplicações de análise de dados trabalham no processamento offline dos dados coletados, criando modelos genéricos que podem ser aplicados aos dados coletados no futuro. Por fim, as aplicações de tempo real para tomada de decisões realizam análises dos dados e geram uma resposta de acordo com as necessidades específicas da aplicação.

Conforme mencionado anteriormente, a IoT é amplamente aplicada em diversas áreas. Segundo Ngu *et al.* (2017) existem exemplos de aplicações em cidades inteligentes, casas inteligentes, ambientes inteligentes e na manufatura. Além disso, Gupta e Quamara (2020) também destaca outras áreas de aplicações, como saúde, logística e varejo, gerenciamento de energia na indústria e transporte inteligente.

De acordo com a Fortune Business Insights (2020) o mercado global de IoT captou em torno de 250,72 bilhões de dólares até 2019 e é previsto atingir a marca de 1.439,19 bilhões de dólares até 2027. Esses números refletem o crescimento exponencial e o enorme potencial econômico da industria de IoT.

De acordo com Lueth (2020) o número de dispositivos conectados atingiu a marca de 9,5 bilhões no final de 2019, superando a estimativa de 8,3 bilhões para esse período. Essa expansão substancial mostra o rápido ritmo de adoção e integração dos dispositivos IoT em várias esferas da vida cotidiana.

Esses dados evidenciam a magnitude do crescimento e o potencial contínuo da indústria de IoT. O investimento em tecnologias de IoT continua a crescer à medida que mais empresas e setores reconhecem os benefícios e as oportunidades oferecidas por essa revolucionária rede de dispositivos conectados.

2.3 Camadas de Rede

Os projetistas de redes adotam uma abordagem de divisão em camadas para os protocolos, em que cada camada pode ser composta por hardware, software ou uma combinação de ambos. Essa estrutura organizada permite a criação de protocolos de rede eficientes e escaláveis. Para a Internet, os protocolos de rede são divididos em cinco camadas principais: física, enlace, rede, transporte e aplicação, conforme mencionado por (KUROSE *et al.*, 2013). Essa divisão em camadas facilita o desenvolvimento, a implementação e a manutenção de sistemas de comunicação em rede, além de promover a interoperabilidade entre diferentes dispositivos e serviços.

A camada de aplicação abriga uma variedade de aplicações de redes e seus respectivos protocolos. Nesse nível, são utilizados diversos protocolos essenciais como HTTP, SMTP e o FTP. O cerne dessa camada reside na capacidade de desenvolver programas que possam interagir de forma eficiente, como por exemplo, nas arquiteturas cliente-servidor e peer-to-peer (P2P).

A camada de transporte é responsável por transportar mensagens entre as aplicações do lado do cliente e do servidor. Esse transporte ocorre por meio de dois protocolos principais: TCP (Transmission Control Protocol) e UDP (User Datagram Protocol). O TCP oferece um serviço orientado a conexão, garantindo a transferência confiável de dados. Por outro lado, o UDP é um serviço não orientado a conexão, permitindo a transferência de dados sem garantia de confiabilidade. No entanto, o UDP possui a vantagem de proporcionar uma maior taxa de transferência de dados, priorizando a vazão em detrimento da confiabilidade. Essa distinção entre TCP e UDP possibilita a adaptação do transporte de acordo com as necessidades específicas de cada aplicação.

A camada de rede desempenha um papel crucial no transporte de pacotes (datagramas) de hospedeiro para outro, garantindo que os pacotes encontrem seus destinatários corretos. Durante o processo de encaminhamento, os roteadores recebem pacotes e devem determinar o enlace de saída apropriado. Por sua vez, o roteamento envolve a tarefa de encontrar a rota ou caminho mais eficiente para que os pacotes fluam de um remetente a um destinatário.

Na camada de enlace estão os roteadores que enviam os datagramas entre a origem e o destino cujos componentes são chamados de nós. Normalmente um nó pode ser um hospedeiro, roteador, comutador e pontos de acesso Wi-Fi. E os canais de comunicação que conectam os nós adjacentes aos caminhos de comunicação podem ser chamados de enlaces. O processo de comunicação e transporte de um datagrama de um hospedeiro de origem a um hospedeiro de destino ocorre em enlaces individuais no caminho fim a fim.

A camada física é responsável por movimentar os bits de maneira individual e os protocolos de comunicação desta camada dependem da camada de enlace e de seu meio de transmissão (fibra óptica, cabo coaxial, por exemplo).

2.4 Protocolos da Camada de Aplicação

Os protocolos desempenham um papel fundamental na transação de dados em aplicações de IoT. São eles que estabelecem a forma como os dados serão transmitidos na rede e os quais recursos que estarão disponíveis. Na camada de aplicação estão presentes os softwares desenvolvidos para solucionar problemas específicos. Ao criar um software, o programador determina qual protocolo a aplicação utilizará, juntamente com outras características relacionadas à troca de mensagem entre diferentes sistemas. A escolha adequada do protocolo e sua configuração correta são essenciais para garantir uma comunicação eficiente, segura e compatível entre os dispositivos na Internet das Coisas.

Alguns dos protocolos utilizados na camada de aplicação são detalhados abaixo. Um deles é o HTTP, responsável pela transferência de hipertexto na Web, que permite a comunicação entre clientes e servidores através de requisições e respostas. Apresentamos também o MQTT, um protocolo amplamente utilizado para a troca de mensagens em ambientes de IoT.

2.4.1 HTTP

O Protocolo de Transferência de Hipertexto (HTTP) foi proposto para a camada de aplicação e surgiu com o objetivo de ser um protocolo escalável, colaborativo para sistemas de informação hipermídia (The Internet Society, 1999). Ele permite o uso de um conjunto aberto de cabeçalhos que identificam o propósito de uma solicitação baseado nas referências fornecidas pelo URI, URL e URN para identificação de recursos.

The Internet Society (1999) define o HTTP como um protocolo genérico que pode ser usado para a comunicação entre agentes de usuários e *proxies/gateways* para outros sistemas na Internet, incluindo suporte aos protocolos SMTP, NNTP, FTP, Gopher e WAIS. O HTTP usa o protocolo TCP como seu protocolo de transporte e não armazena estado, portanto, ele é um protocolo sem estado (*stateless*) (KUROSE *et al.*, 2013).

Conexões HTTP podem ocorrer de duas formas: não persistentes e persistentes. Em conexões não persistentes o cliente HTTP inicia uma conexão TCP para um servidor e envia uma mensagem com requisição HTTP para o servidor, o servidor HTTP recebe a requisição, extrai o objeto, encapsula a mensagem de resposta e envia ao cliente. O processo HTTP, então encerra a conexão TCP (KUROSE *et al.*, 2013). Toda vez que uma requisição é realizada, uma conexão não persistente deve-se abrir, e em seguida uma conexão com o servidor que é encerrada ao final da operação.

Um modelo de troca de mensagens persistente abre uma conexão com o servidor e mantém a conexão aberta para várias requisições. A conexão se encerra quando não é usada por um período de tempo.

2.4.2 MQTT

O Transporte de telemetria de enfileiramento de mensagens (MQTT) é um protocolo de transporte de mensagens *publish/subscribe* para modelos cliente-servidor, além de ser um modelo leve, aberto, simples e projetado para ser facilmente implementado (OASIS, 2019). Esse modelo é ideal para muitas situações como: Internet das Coisas e comunicações *machine-to-machine* (M2M).

O MQTT executa no protocolo TCP/IP, ou, em protocolos de rede que fornecem conexões bidirecionais ordenadas e sem perdas. Além disso, ele apresenta recursos como distribuição de mensagem de um para muitos com o uso do *publish/subscribe*. O transporte de mensagens independe do conteúdo de carga útil, três diferentes tipos de qualidade de serviço, pequena

sobrecarga no transporte e trocas de mensagens para reduzir o tráfego de rede e um mecanismo de aviso para as partes interessantes quando ocorrem desconexões.

Um modelo de troca de mensagens usando o MQTT é apresentado na Figura 1. Neste modelo há os *publishers* responsáveis por gerar novas mensagens para o servidor do MQTT (Broker) que coordena as requisições, os *subscribers* que escutam as publicações em tópicos específicos. Tais tópicos possuem nomes únicos e quando um *publisher* envia um dado para determinado tópico os *subscribers* que estão escutando aquele tópico recebem a mensagem publicada.

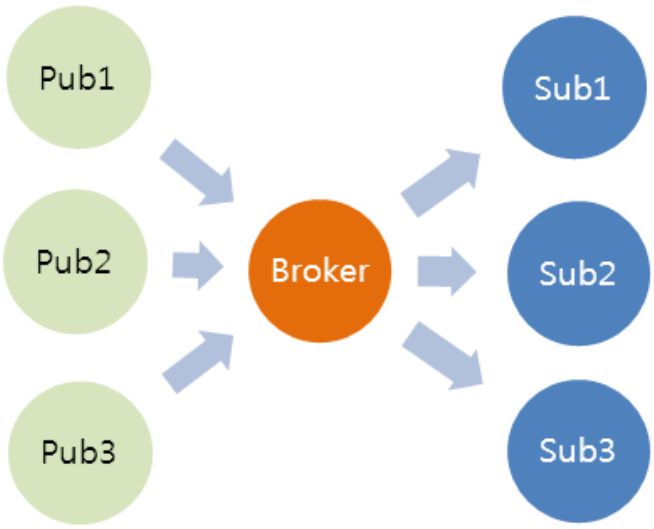

Figura 1 – Protocolo MQTT

2.5 Protocolos da Camada de Transporte

O Protocolo de Controle de Transmissão (TCP) e o Protocolo de Datagrama de Usuário (UDP) pertencem a camada de transporte. Esses protocolos são usados para o transporte de dados, cuja diferença entre eles está na confiabilidade da forma de transmissão do dado.

O TCP foi projetado para ser utilizado como um protocolo *host-a-host* altamente confiável para comutação de pacotes (The Internet Society, 1981). Portanto, o TCP é um protocolo seguro para a transferência de dados entre processos que executem em *hosts* diferentes.

O UDP fornece um modo de programas aplicativos enviem mensagens envie mensagens a outros programas aplicativos com o mínimo de mecanismos de protocolo (The Internet Society, 1980). Isso quer dizer que o protocolo reduz o nível de confiabilidade ao enviar as mensagens entre aplicações.

A diferença na confiabilidade do dado enviado no TCP e UDP é importante. Em razão da confiabilidade que o TCP proporciona ele se torna um protocolo mais lento no envio de

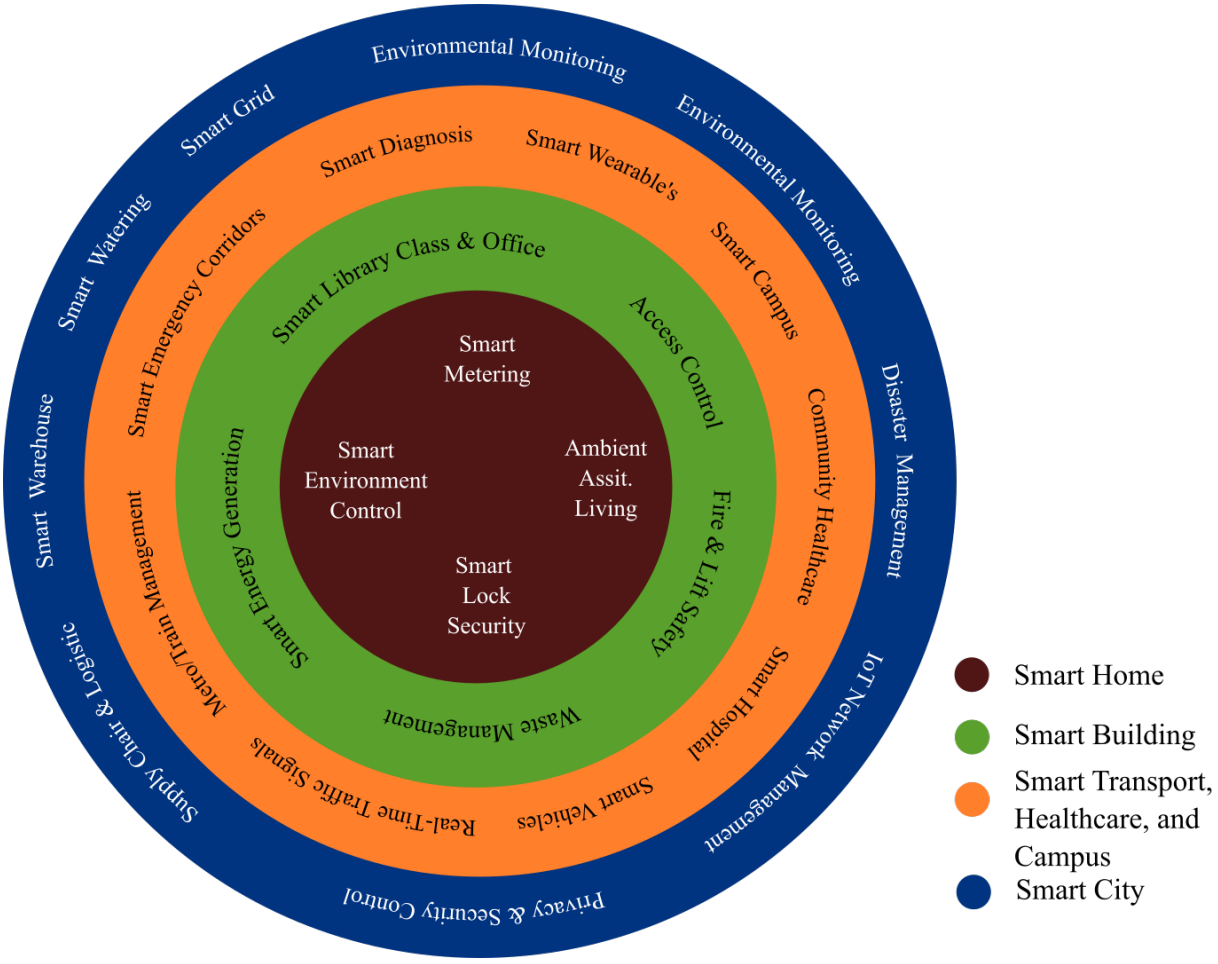

Figura 2 – Aplicações de pequena para grande escala em IoT adaptado de Sunhare, Chowdhary e Chattopadhyay (2020)

mensagens, em contrapartida o UDP não garante que a mensagem vai ser entregue, mas fornece um ambiente mais rápido no envio de mensagens.

2.6 Aplicações de IoT

Em um estudo realizado por Sunhare, Chowdhary e Chattopadhyay (2020) as aplicações de IoT foram classificadas de pequeno à grande porte. A Figura 2 apresenta as aplicações de IoT destacando que as aplicações menores são utilizadas na construção de aplicações maiores.

São apresentados quatro blocos distintos: casas inteligentes, edifícios inteligentes, saúde e transporte inteligentes e cidades inteligentes. Os blocos internos servem como estruturas para a criação dos blocos externos. A análise dessa imagem é interessante, por que, expressa a necessidade de diversas pequenas aplicações de IoT que comunicam entre si para a criação de estrutura maiores. Essas características demonstram a heterogeneidade e a necessidade de pensar em IoT como pequenos aglomerados que se comunicam e geram aplicações mais complexas.

As subseções a seguir apresentam observações acerca de aplicações de pequena, média e

grande escala para IoT, enquanto explica as diversas subaplicações de cada uma destas escalas.

Casas inteligentes serão tratadas como aplicações de pequena escala. Edifícios, saúde e transporte inteligentes serão tratados como aplicações de médio porte. E por fim, cidades inteligentes serão tratadas como aplicação de grande porte.

2.6.1 Pequena Escala

Aplicações de pequena escala para IoT não necessitam de muito poder computacional. Os dados da aplicação podem ser gerenciados por um único dispositivo de última geração (*High End Device* - HED). Conforme apresentado por Bansal e Kumar (2020), esse tipo de HED é composto por computadores de placas únicas que possuem um grande número de recursos computacionais como CPU, RAM, memória flash, etc.

Normalmente são utilizados como suporte para esses HEDs microcontroladores de menor capacidade computacional, sensores e atuadores. Os microcontroladores de menor capacidade são responsáveis por orquestrar a transmissão de dados dos sensores e atuadores. Os sensores são responsáveis por receber dados gerados no ambiente e os atuadores executam ações.

2.6.1.1 Casas Inteligentes (Smart Homes)

Casas inteligentes podem ser definidas como eletrodomésticos e dispositivos controlados por meios remotos (GAIKWAD; GABHANE; GOLAIT, 2015). Já Park *et al.* (2018) definiram casas inteligentes como um conjunto de tecnologias orientadas ao ser humano que fornecem um ambiente de rede para conectar ambientes e aplicativos. Portanto, pode-se definir casas inteligentes como uma casa monitorada em tempo real no qual o ser humano pode realizar acesso remoto e fazer controle dos dispositivos nela instalados.

2.6.2 Média Escala

Conforme definido anteriormente, aplicações de média escala são projetadas sob aplicações menores. Vamos abordar nesta sub-seção edifício, transporte e saúde inteligente.

Para edifícios e saúde inteligente um análogo com casas inteligentes é facilmente observável. Cada cômodo ou bloco de um edifício/hospital inteligente possui propriedades semelhantes a uma casa. Dessa forma, vários blocos/cômodos são uma superestrutura de uma casa inteligente que necessitam ser gerenciados com mais recursos computacionais.

Já em transporte inteligente esse análogo não é fácil de ser realizado. Mas se observarmos estrutura e componentes físicos necessários para gerenciar em tempo real esse tipo de aplicação basta substituirmos os eletrodomésticos por sensores específicos para mapear a localização do veículo, por exemplo. Além disso, para realizar o gerenciamento do veículo enquanto ele se desloca pela cidade é necessária uma infraestrutura que consiga fornecer comunicação entre os

dispositivos e sensores instalados no veículo e uma rede que consiga transmitir os dados para a Internet.

2.6.2.1 Edifícios/Construções Inteligentes (Smart Buildings)

Construções inteligentes (*Smart Buildings*) são um elemento central na melhoria de infraestruturas e cidades, mas, acima de tudo, na melhoria do conforto dos residentes (DAISSAOUI *et al.*, 2020). Ainda, segundo Daissaoui *et al.* (2020), construções inteligentes permitem maior eficiência energética, controlam aspectos de segurança, levam em consideração aspectos de conforto, qualidade de vida e serviços.

A junção de IoT e construções inteligentes resultou em diversas aplicações interessantes, entre elas, gerenciamento de energia, rastreamento de localização de ocupantes e recursos, melhoria do conforto interno e gestão de instalações (DAISSAOUI *et al.*, 2020).

2.6.2.2 Transporte Inteligente (Smart Transport)

Transporte inteligente foca na economia de energia, desenvolvimento econômico e qualidade de vida usando tecnologias inovadoras e além disso, pode abranger a criação de aplicações, avanços tecnológicos e abordagens que podem construir o conceito de desenvolvimento urbano no futuro (BIYIK, 2019).

Zhang, Yu e Zhai (2011) apresentam uma ideia semelhante ao afirmar que engarrafamentos levam a perdas econômicas, agravam crises de energia e o transporte urbano está diretamente relacionado a qualidade de vida urbana.

Portanto, para a existência de uma infraestrutura funcional em cidade inteligente é necessário um sistema de transporte funcional, que consiga reduzir o tráfego, o consumo de combustível e energia, enquanto proporciona uma melhor qualidade de vida para a população.

2.6.2.3 Saúde Inteligente (Smart Healthcare)

Saúde inteligente pode ser definida como a integração entre pacientes e médicos em uma plataforma comum para monitoramento inteligente da saúde, pela análise dia-a-dia das atividades humanas (LOHACHAB, 2019). Ou seja, saúde inteligente está vinculada a coleta de informações dos pacientes que são enviadas para médicos responsáveis de forma que estes consigam visualizar as informações coletadas dos pacientes e fornecer melhores tratamentos, medicamentos e assistências.

Exemplos de aplicações de saúde inteligente podem ser vistas em Thaduangta *et al.* (2016) que se propõe em criar uma plataforma para monitoramento de idosos e em (CAI *et al.*, 2019) são apresentadas algumas soluções existentes para saúde inteligente e processos para a tomada de decisão.

2.6.3 Grande Escala

Com pequena e médias aplicações conectadas a criação de uma aplicação maior, como, uma cidade inteligente se torna possível.

2.6.3.1 Cidades Inteligentes (Smart Cities)

Cidades inteligentes interligam diferentes sistemas ciberfísicos (*Cyber Physical Systems* - CPS) por meio da troca de dados brutos ou pré-processados permitindo uma melhora na vida dos habitantes da cidade (PULIAFITO *et al.*, 2021). Um sistema ciberfísico é aquele cujos os elementos computacionais colaborativos controlam entidades físicas no ambiente.

Para Kirimtat *et al.* (2020) cidades inteligentes são uma rede crescente de sensores digitais, dispositivos inteligentes e eletrodomésticos inteligentes utilizados para a melhora da condição de vida de seus habitantes. Cidades se tornam mais inteligentes com a adoção de tecnologias como Internet das Coisas, *big data* e a adoção de tecnologias em nuvem.

2.7 Microcontroladores

Microcontroladores são dispositivos eletrônicos compactos que contêm um processador, memória e periféricos integrados em um único chip. Esses componentes são projetados para executar tarefas específicas e controlar dispositivos ou sistemas em uma variedade de aplicações. Com seu tamanho reduzido e baixo consumo de energia, os microcontroladores são amplamente utilizados em eletrônica embarcada, sistemas de controle industrial, dispositivos médicos, automação residencial e muitas outras áreas.

Uma das principais vantagens dos microcontroladores é a sua capacidade de integração de recursos em um único chip. Eles possuem memória programável, permitindo o armazenamento de código de controle e dados de aplicação. Além disso, os microcontroladores oferecem uma variedade de periféricos integrados, como portas de entrada e saída (I/O), interfaces de comunicação (como UART, SPI e I2C), conversores analógico-digitais (ADC) e timers. Esses recursos tornam os microcontroladores altamente flexíveis e adaptáveis para atender às necessidades específicas de um projeto.

Os microcontroladores também são conhecidos por sua eficiência energética, o que os torna ideais para aplicações que exigem baixo consumo de energia ou operação em baterias. Eles podem ser programados e reprogramados para executar diferentes tarefas e se adaptar a diferentes requisitos de aplicação. Além disso, existem diversas plataformas e linguagens de programação disponíveis para o desenvolvimento de aplicações em microcontroladores, facilitando a criação de soluções personalizadas e de alto desempenho. Com sua versatilidade e poder de processamento em um pacote compacto, os microcontroladores desempenham um papel fundamental em inúmeras aplicações eletrônicas.

2.7.1 ESP-32

O ESP32 é um microcontrolador de baixo consumo de energia baseado na arquitetura de 32 bits da família Xtensa LX6 da empresa Tensilica. Ele é projetado para aplicações de Internet das Coisas (IoT) e possui um alto grau de integração de componentes em um único chip. O ESP32 incorpora um processador dual-core, que permite a execução simultânea de tarefas e oferece maior desempenho em comparação com microcontroladores de núcleo único.

O microcontrolador ESP32 possui uma ampla gama de periféricos integrados, incluindo portas de entrada e saída (I/O) digitais e analógicas, interfaces de comunicação, como Universal Asynchronous Receiver/Transmitter (UART), Serial Peripheral Interface (SPI) e Inter-Integrated Circuit (I2C), além de módulos Wi-Fi e Bluetooth. Esses recursos permitem a conectividade sem fio e a comunicação eficiente com outros dispositivos e sistemas.

Além disso, o ESP32 oferece suporte a diversos protocolos de rede, como IPv4 e IPv6, e possui recursos avançados de segurança, incluindo criptografia e autenticação. Ele também é compatível com uma variedade de ambientes de desenvolvimento, incluindo a linguagem de programação C/C++ e frameworks como o Arduino.

O ESP32 é amplamente utilizado em projetos de IoT, devido à sua versatilidade, baixo consumo de energia e capacidade de processamento. Ele oferece uma solução eficiente para aplicações que exigem conectividade, processamento de dados em tempo real e interação com o ambiente físico. Com sua arquitetura avançada e recursos integrados, o ESP32 é uma opção popular para a criação de dispositivos IoT de alta performance.

2.8 Sistemas de Mensageria

Sistemas de mensageria são componentes fundamentais na arquitetura de sistemas distribuídos, permitindo a comunicação assíncrona e confiável entre diferentes partes de um sistema. Esses sistemas atuam como intermediários eficientes, garantindo o envio e a entrega de mensagens entre os diversos componentes de forma escalável e resiliente. Com a utilização de sistemas de mensageria, é possível desacoplar os componentes do sistema, permitindo que eles se comuniquem de forma independente, o que resulta em maior flexibilidade, modularidade e facilidade de manutenção.

Um exemplo popular de sistema de mensageria é o Apache Kafka. Ele se destaca por sua arquitetura distribuída, alta capacidade de processamento e escalabilidade. O Kafka utiliza o modelo de publicação e assinatura, onde os produtores enviam mensagens para tópicos específicos, enquanto os consumidores se inscrevem nos tópicos de interesse para receber e processar as mensagens. Com seus recursos avançados de replicação, tolerância a falhas e armazenamento durável em log, o Apache Kafka é amplamente utilizado em cenários de streaming de dados em tempo real, processamento de eventos, integração de sistemas e análise

de dados em escala.

2.9 Considerações Finais

Este capítulo apresenta o referencial teórico necessário para acompanhar o desenvolvimento deste trabalho. Apresentamos um pouco sobre o conceito de IoT, sobre alguns protocolos da camada de aplicação, sobre os protocolos da camada de transporte, apresentamos uma forma de visualização de aplicações de IoT de acordo com a escala da aplicação, apresentamos os conceitos de microcontroladores e, por fim, apresentamos o conceito de sistema de mensageria.

CAPÍTULO

3

TRABALHOS RELACIONADOS

3.1 Considerações Iniciais

Neste capítulo apresentamos o processo utilizado para a seleção de trabalhos e uma visão geral de como as arquiteturas de cada trabalho bibliográfico funciona. Faz-se algumas relações entre as arquiteturas e apresenta-se uma conclusão de quais características são mais evidentes que servirão de base para este projeto de mestrado.

3.2 Revisão da Literatura

A revisão bibliográfica foi realizada nas bases de dados do *Web of Science*, consultando artigos com a palavra chave "arquitetura", "aplicação", "implementação"e "iot", além de palavras, chave conforme apresentado na Figura 2. A *string* de consulta é apresentada abaixo:

"iot"and "architecture"and ("application"or "implementation") and ("smart city"or "smart transport"or "smart healthcare"or "smart building"or "smart home"or "smart environment control"or "smart metering"or "ambient assistance living"or "smart lock security"or "access control"or "fire safety"or "lift safety"or "waste management"or "smart energy generation"or "smart library"or "smart office"or "smart diagnosis"or "smart wearables"or "smart campus"or "community healthcare"or "smart hospital"or "smart vehicles"or "real-time traffic signals"or "metro management"or "train management"or "smart emergency corridors"or "environmental monitoring"or "disaster management"or "IoT network management"or "privacy control"or "security control"or "suplly chair"or "logistic"or "smart warehouse"or "smart watering"or "smart grid")

Foram retornados vários resultados, dentre os quais foram filtrados os trabalhos publicados entre 2016-2021 e aqueles pertencentes as áreas de "Ciências da Computação"e "Engenharia". Dessa

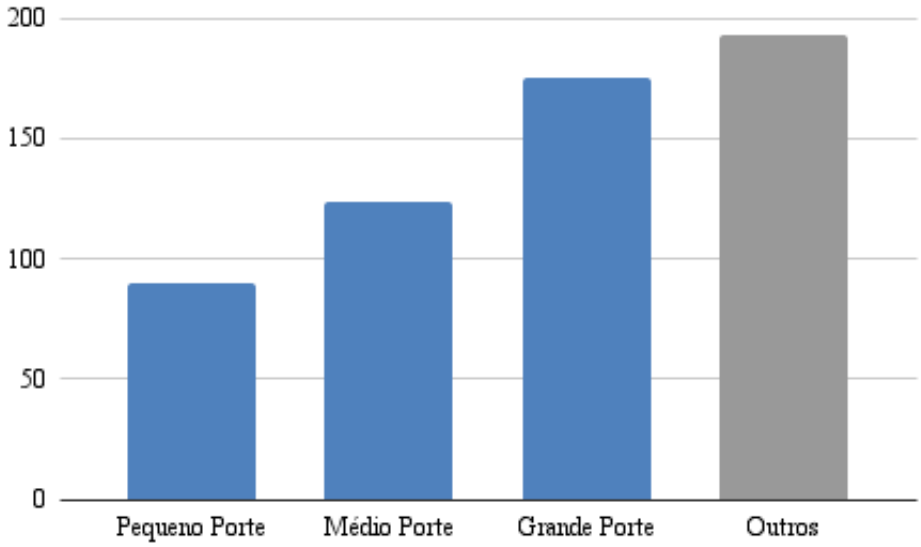

Figura 3 – Relação dos trabalhos depois da primeira seleção

forma, ao final da consulta na escrita da proposta foram retornados um total de 581 resultados.

As seguintes abordagens foram utilizadas com o intuito de selecionar a bibliografia utilizada.

- Separação dos artigos/periódicos em categorias de acordo com as aplicações de IoT de pequeno, médio e grande porte;
- Desenvolvimento de aplicações utilizando a arquitetura proposta;
- Considerar os mais citados e a classificação das revistas e períodos aos quais eles foram publicados; e
- Análise de resumos.

O gráfico apresentado na Figura 3 apresenta os resultados após a primeira filtragem, relacionando os artigos entre as categorias de pequenas, média e grandes aplicações. Houve 90 artigos separados em pequena escala, 124 relacionados a média escala e 175 relacionados a grande escala. Os artigos restantes não foram enquadrados em nenhuma dessas escalas por que não se encaixaram em uma dessas categorias ou não têm relação com o objetivo desse trabalho, a qual está representada em cinza no gráfico.

A segunda filtragem específica de cada categoria separou os trabalhos que apresentam algum tipo de arquitetura ou infraestrutura daqueles que não apresentam. Foi analisado o resumo dos trabalhos e a ocorrência dessa informação. Após essa etapa sobraram 32 trabalhos de pequena escala, 45 relacionado a média escala e 43 de grande escala. O gráfico apresentado na Figura 4 apresenta essa distribuição.

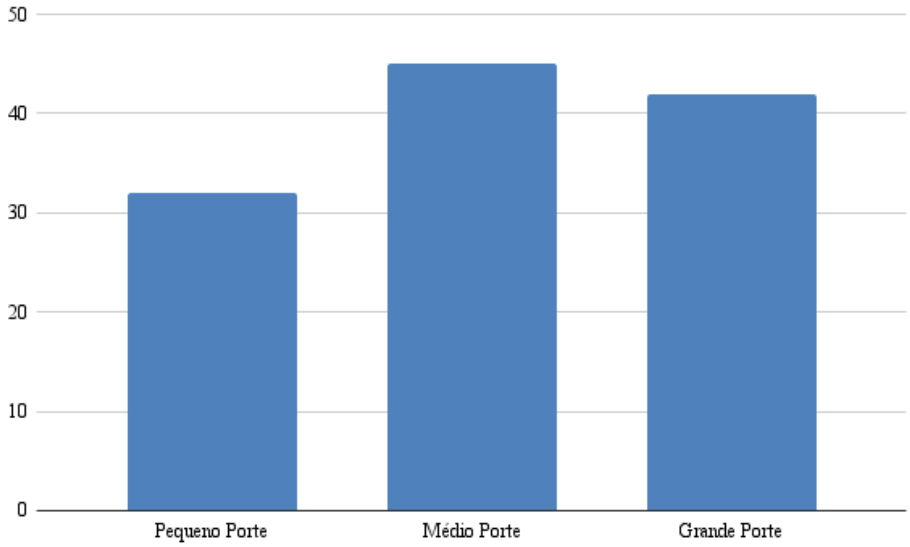

Figura 4 – Relação dos trabalhos depois da segunda seleção

A próxima filtragem contou com a classificação dos artigos conforme a avaliação do periódico ao qual foi publicado e a sua quantidade de citações. Essa classificação é importante para selecionar os melhores trabalhos para cada tamanho arquitetural. Dentre esses artigos foram selecionados os 6 que tiveram mais citações e foram publicados em conferências mais relevantes, segundo o site Novo Qualis CC [1]. Alguns trabalhos não tiveram suas conferências avaliadas nessa aplicação, mas foram consideradas mesmo assim, devido a quantidade de citações.

Por fim, a partir da avaliação dos resumos foram selecionados os artigos citados nos próximos tópicos.

3.2.1 Arquiteturas de Pequena Escala

Neste tópico apresentaremos os trabalhos de pequena escala selecionados conforme a abordagem apresentada. Esses trabalhos estão relacionados principalmente a casas inteligentes. Serão discutidas as arquiteturas apresentadas em cada um desses trabalhos, além dos componentes de hardware, software e rede empregados nessas soluções.

A Tabela 8 no Apêndice A apresenta um resumo desses trabalhos e quais aplicações foram desenvolvidas.

Malche e Maheshwary (2017) apresentaram uma plataforma para a construção de casas inteligentes para IoT. É apresentado uma arquitetura FLIP (*Frugal Labs IoT Platform*) que é implementado em uma casa inteligente. Para essa aplicação de casa inteligente foram desenvolvidos módulos de alerta, monitoramento, controle e inteligência para aplicações de iluminação inteligente, eletrodomésticos inteligentes, detecção de intrusão e detecção de fumaça e gás. A

[1] https://ppgcc.github.io/discentesPPGCC/pt-BR/qualis/

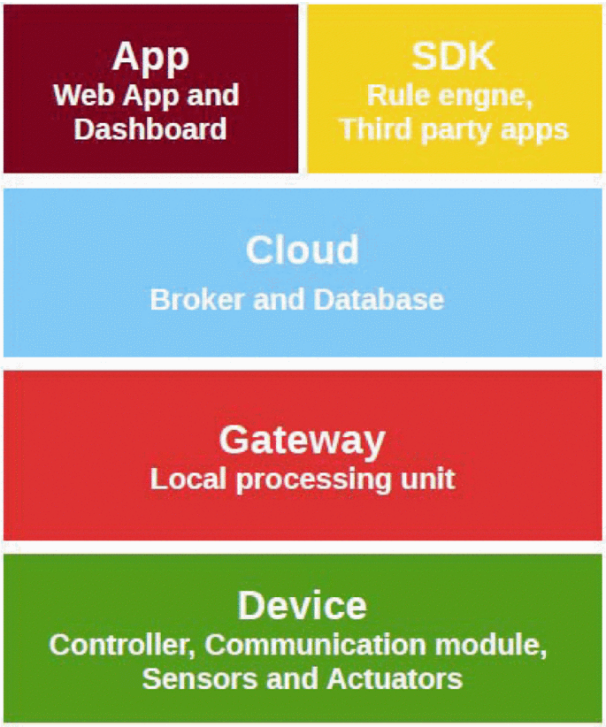

Figura 5 – Arquitetura proposta por Malche e Maheshwary (2017)

arquitetura proposta é apresentada na Figura 23.

Nas camadas inferiores temos os dispositivos físicos com menor capacidade de processamento, como os controladores, sensores, atuadores, os módulos de comunicação e controladores. Nessas camadas inferiores foram utilizados componentes como: um módulo *WiFi* e um módulo *bluetooth*. O *gateway* é unidade de processamento e nesse caso foi utilizado o *Raspberry PI 3*. A camada de nuvem consiste em um *broker* e um banco de dados para armazenamento da informação e nessa camada temos tecnologia como MQTT, MongoDB e NodeJS. Por fim, são apresentadas camadas de apresentação da informação (*dashboard*) e de aplicações para terceiros.

Outra plataforma de IoT para um sistema doméstico inteligente usando *Web-of-Objects* [2] e arquitetura em nuvem é proposta por Iqbal *et al.* (2018a). É apresentado um *gateway* na tecnologia *Raspberry PI* para conectar diferentes tecnologias e protocolos. São coletados dados de eletrodomésticos que são gerenciados na WEB utilizando o protocolo *REST/RESTFul* e um servidor em nuvem para armazenar as informações coletadas nestes diversos ambientes. Além disso, é utilizado a tecnologia *ZigBee* no controle das caixas d'água e segurança na porta usando uma câmera com Protocolo de Rede (IP). Já em Iqbal *et al.* (2018b) é apresentada uma solução para o melhor uso dos eletrodomésticos e da energia da casa. Um diferencial das arquiteturas anteriormente apresentadas é o uso do ecossistema *Hadoop* para melhorar a eficiência e diminuir o tempo de processamento do dado.

[2] Padrões que descreve um conjunto de padrões do W3C para resolver os problemas de interoperabilidade de diferentes plataformas de Internet das Coisas (IoT) e domínios de aplicativo.

É interessante a comparação entre os dois trabalhos Iqbal *et al.* (2018a) e Iqbal *et al.* (2018b). Pois, o primeiro está mais preocupado com o acúmulo dos dados e a exibição de status dessas casas inteligentes e o segundo está preocupado com o processamento do dado e entregar valor com esse dado que é processado. A diferença principal consiste na adição da ferramenta *Hadoop* que proporciona essa função de tratamento dos dados em tempo real e diminuição do tempo de latência.

Petnik e Vanus (2018) integraram um sistema que utiliza o protocolo *KNX* com uma casa inteligente usando uma plataforma de IoT baseada na nuvem. É utilizada a camada baseada na nuvem como a interface de integração do sistema entre os outros dois componentes. Por fim, são aplicadas técnicas de interface de linguagem natural para extrair informações dos dados e apresentar essas informações em uma linguagem mais próxima do usuário final.

A arquitetura proposta por Amadeo *et al.* (2017) está vinculada à nuvem computacional para a resolução do problema de heterogeneidade, comunicações em baixa latência e robustez. Propõe-se nesse trabalho uma rede centrada na informação e uma camada de névoa computacional (*Fog Computing*). A proposta arquitetural neste caso foca no processamento mais próximo dos dispositivos de geração de dados de forma que a carga de trabalho do processamento é dividida entre as camadas de nuvem e névoa.

Por fim, todas as cinco arquiteturas apresentadas para pequena escala apresentam camadas e características em comum. Essas características em comum estão relacionadas aos sensores, *gateways* de controle e infraestrutura na nuvem para o controle dessas aplicações. Diferenças se encontram principalmente na divisão das camadas e se foram ou não utilizados algoritmos para fornecer inteligencia para o sistema.

3.2.2 Arquiteturas de Média Escala

Neste tópico apresentaremos os trabalhos de média escala selecionados conforme a abordagem apresentada. As aplicações aqui são mais heterogêneas do que as apresentadas em pequena escala. Apresentamos arquiteturas relacionadas à saúde inteligente, gerenciamento de controle de acesso e gestão de resíduos.

Rahmani *et al.* (2018) abordaram uma solução para saúde baseada em IoT. Nesta solução os *gateways* estão localizados na borda computacional da rede e oferecem alto nível de armazenamento como armazenamento local, processamento de dados em tempo real e mineração de dados embutida. O objetivo desse trabalho é apresentar uma solução para distribuir geograficamente a camada intermediária para explorar o conceito de computação em névoa para a resolução de problemas na área da saúde. Concluindo afirma-se que uma solução bem implementada com a arquitetura proposta oferece suporte para integração de diversos sistemas para o monitoramento da saúde.

Catarinucci *et al.* (2015) apresenta uma solução para sistemas de hospitais inteligentes.

Esta solução destaca elementos de rede e transferência de informação como RFID, redes de sensores sem fio, COaP, 6LoWPAN e REST. Desta forma, os dados são centralizados em uma infraestrutura para o consumo via REST API's. Em comparação com Rahmani *et al.* (2018) essa arquitetura é mais direta, porém, o problema desse tipo de arquitetura é o gargalo nos *gateways* e no tratamento precoce nos dados.

Outra solução para saúde inteligente é proposta por Khoi *et al.* (2015). Essa proposta se baseia no protocolo CoAP para transferência de dados. Uma camada de névoa computacional realiza o processamento dos dados, que utiliza o *middleware openHAB*. Por fim, há mais duas camadas, uma de transferência de dados na rede e a outra de nuvem para armazenamento, processamento e monitoramento dos dados.

A arquitetura proposta por Ouaddah, Kalam e Ouahman (2016) está relacionada ao gerenciamento do controle de acesso com o uso de *blockchains*. Como prova de conceito foi implementado um *blockchain* local em *Raspberry PIs* para conceder, obter, delegar e revogar acessos. Na arquitetura uma camada de controle do *blockchain* foi adicionada entre o *gateway* e o gerenciamento que ocorre geralmente na nuvem.

Wen *et al.* (2018) em seu trabalho mostra outra vertente do uso de IoT ao realizar a gestão de resíduos. Nessa arquitetura a principal característica é o controlador de bordo presente nos caminhões de coleta de lixo. Diferente das soluções arquiteturais apresentadas, essa solução precisa se manter em funcionamento enquanto o caminhão se desloca e sua localização muda no espaço. A solução foi o uso de dispositivos de transmissão *GPRS* e 3G, além de controlar a localização com o uso de *GPS*.

Bharadwaj, Rego e Chowdhury (2016) semelhante a Wen *et al.* (2018), apresenta uma arquitetura de gestão de resíduos. No caso de Bharadwaj, Rego e Chowdhury (2016) apresentou-se uma arquitetura que usa a tecnologia Longo Alcance (LoRa). Essa tecnologia permite uma comunicação de longa distância. Dessa forma, o dado é coletado em lixeiras, enviado a longas distâncias com a tecnologia LoRa. Depois, o dado é recebido pelo protocolo MQTT que extrai informação e o armazena. Por fim, é apresentado um portal *WEB* para a visualização das informações coletadas.

Em comparação com as aplicações de pequena escala observa-se algumas diferenças. A primeira delas é a distância e as tecnologias usadas para contornar esse problema, essa característica pode ser vista em Wen *et al.* (2018) e Bharadwaj, Rego e Chowdhury (2016). A segunda está relacionada a necessidade de processamento do dados em diversas etapas da arquitetura e não apenas na nuvem como pode ser visto em Rahmani *et al.* (2018) e Khoi *et al.* (2015).

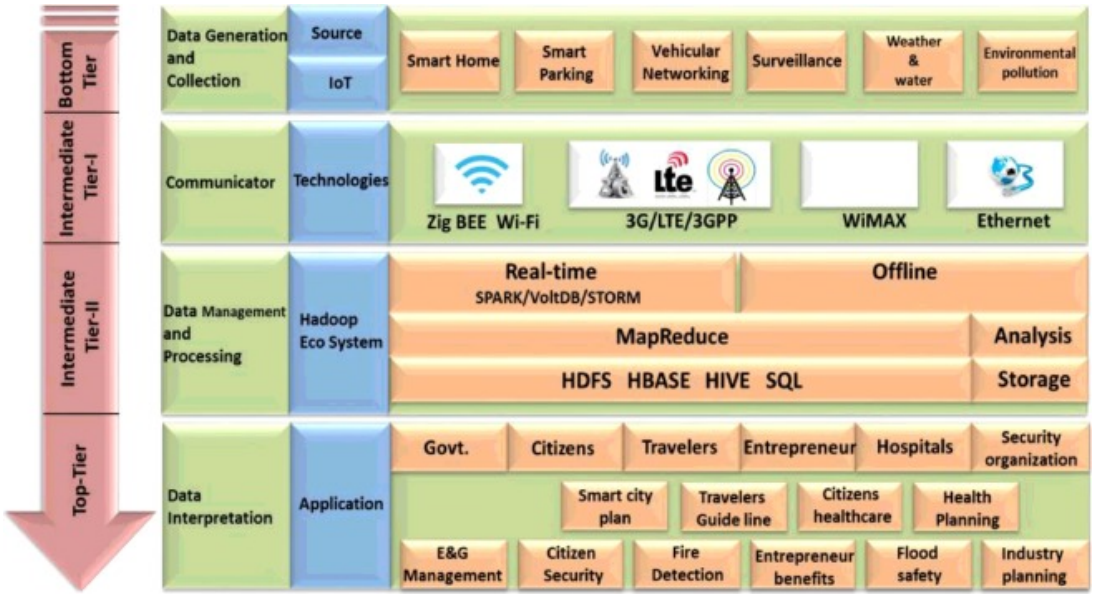

Figura 6 – Arquitetura proposta por Rathore *et al.* (2016)

3.2.3 Arquiteturas de Grande Escala

Neste tópico apresentaremos os trabalhos de grande escala selecionados conforme a abordagem apresentada. A principal característica dessa categoria se encontra na forma como se trata o processamento de dados. Em arquiteturas de grande escala o foco é no uso de ferramentas para o gerenciamento dos dados gerados sendo empregadas ferramentas de Big Data e processamento em tempo real.

Rathore *et al.* (2016) apresentaram em seu trabalho uma arquitetura que conecta diversas aplicações de IoT para chegar a uma aplicação de cidade inteligente. Nesse trabalho ele propõe uma arquitetura de quatro camadas, sendo a camada inferior responsável por geração de dados e coleta de dados, a camada intermediária 1 responsável pela comunicação entre sensores, *relays*, estações base e a Internet. A camada intermediária 2 é responsável pelo gerenciamento dos dados e processamento com o uso do *framework Hadoop* e a camada superior é responsável pela aplicação e o uso de análise dos dados e dos resultados gerados. A Figura 6 apresenta uma representação visual dessa arquitetura. Essa arquitetura demonstra a heterogeneidade de componentes de uma aplicação de IoT e como diversas aplicações menores colaboram para a criação de aplicações maiores.

O trabalho proposto por Rathore *et al.* (2018) é uma evolução do trabalho proposto em Rathore *et al.* (2016). Nesse trabalho são adicionadas à arquitetura proposta na Figura 6, uma camada de processamento dos dados em tempo real.

Já Montori, Bedogni e Bononi (2018) abordaram o problema de uma arquitetura para cidades inteligentes de outra visão. A visão de uma plataforma que consegue se comunicar com plataformas de terceiros para gerar variedade na coleta de dados e no compartilhamento de serviços. Essa aplicação foi chamada de Internet das Coisas Colaborativa (C-IoT).

Um projeto de *smart grid* é apresentado por Viswanath *et al.* (2016) que propõe uma arquitetura focada em conectar aplicações de gerenciamento de energia usando tecnologias de nuvem, aplicações mobile e *gateways* residenciais. Uma análise em requisitos não funcionais é realizada para medir a qualidade da arquitetura.

Park *et al.* (2019) propõe uma abordagem em Internet das Coisas Cognitiva (CIoT). O foco neste trabalho foi em relação as tecnologias utilizadas para análise de *Big Data* e inteligência artificial.

Os trabalhos de grande escala focam especialmente na coleta de dados de diversas fontes e em como analisar essa enorme quantidade de dados. As arquiteturas desta etapa recebem softwares de *Big Data* para processamento em tempo real dos dados e geração de informação. Por fim, essas aplicações integram diversas outras pequenas aplicações, e tais pequenos microcosmos se juntam para a criação de uma aplicação macro. Essas aplicações micro quando integradas as aplicações macro não perdem suas características individuais, portanto, apresentam desafios, limitações únicas.

3.3 Considerações Finais

Os desafios enfrentados no mapeamento da arquitetura de pequena, média e grande escala mudam de acordo com a escala da aplicação. E arquiteturas maiores procuram solucionar seus próprios problemas arquiteturais e os problemas arquiteturais das aplicações menores que serão integradas.

Este trabalho tem como objetivo determinar blocos genéricos em aplicações de pequena, média e grande escala. Determinar como os dados trafegam em cada uma dessas arquiteturas e implementar uma aplicação de pequena e média escala usando esses modelos para posterior análise, essa abordagem é interessante, por que, dado a premissa que deve-se implementar um ambiente de cidade inteligente em uma cidade física, quais as dificuldades operacionais de aplicações menores que devem ser superadas para se alcançar essa implementação. Modelos arquiteturais genéricos auxilia na tomada de decisões.

Como objetivo específico deste trabalho, o desejo é que por meio de arquiteturas genéricas, seja possível, fazer uma demonstração do fluxo de dados nesse tipo de sistema. Fluxo de dados são os caminhos que o dado trafega desde sua coleta, tratamento, processamento e armazenamento. No meio termo dessa coleta, tratamento, processamento e armazenamento o dado passa por diferentes dispositivos físicos, softwares, protocolos de rede e outros componentes que podem haver nas arquiteturas ou ambientes de IoT.

CAPÍTULO

4

DESENVOLVIMENTO DO MODELO ARQUITETURAL PARA SINTETIZAÇÃO DE FLUXO DE DADOS

4.1 Considerações Iniciais

As arquiteturas apresentadas nos Apêndices B, C, D são as referências para a criação das arquiteturas genéricas que serão apresentadas na seção 4.2. Este mapeamento foi realizado para as arquiteturas de pequeno, médio e grande porte extraindo os principais componentes dessas arquiteturas apresentadas. A partir da extração desses componentes principais, criou-se abstrações capazes de descrever os componentes mínimos para arquiteturas de pequeno, médio e grande porte. Além disso, as abstrações também foram utilizadas para a criação de fluxogramas que demonstram como os dados trafegam nessas redes genéricas.

4.2 Mapeamento das Arquiteturas

Esta seção está organizada da seguinte forma: a Subseção 4.2.1 apresenta os detalhes da arquitetura de pequeno porte e seu respectivo fluxograma. Na Subseção 4.2.2, apresentamos a arquitetura de médio porte, suas diferenças em relação à arquitetura de pequeno porte e seu respectivo fluxograma. Na Subseção 4.2.3, apresentamos a arquitetura de grande porte e sua comparação com a arquitetura de médio porte, juntamente com seu respectivo fluxograma.

4.2.1 Pequena Escala

As arquiteturas de pequena escala descritas no Apêndice B apresentam semelhanças significativas em suas estruturas. Os sensores ou atuadores na camada física se conectam a uma plataforma de gerenciamento dos dados, normalmente *gateways* ou controladores. Essas

plataformas fazem uma conexão com a *Internet* de maneira que os dados são transportados pela internet até servidores na nuvem. Nestes servidores, os dados são tratados e armazenados em bancos de dados.

As diferenças encontradas nessas arquiteturas estão vinculadas aos dispositivos físicos, softwares utilizados, bem como na organização arquitetural com o uso de diferentes protocolos e formas de transferência de informação. Como essas diferenças são em sua maioria tecnológicas, é possível determinar uma abstração arquitetural que não está vinculada a tecnologias específicas.

Conforme as arquiteturas de pequena escala do Apêndice B, propomos uma arquitetura de pequena escala que tem por objetivo ser genérica. A arquitetura proposta pode ser vista na Figura 7. Ela é semelhante à arquitetura proposta por Malche e Maheshwary (2017), porém, com algumas diferenças para torná-la mais genérica.

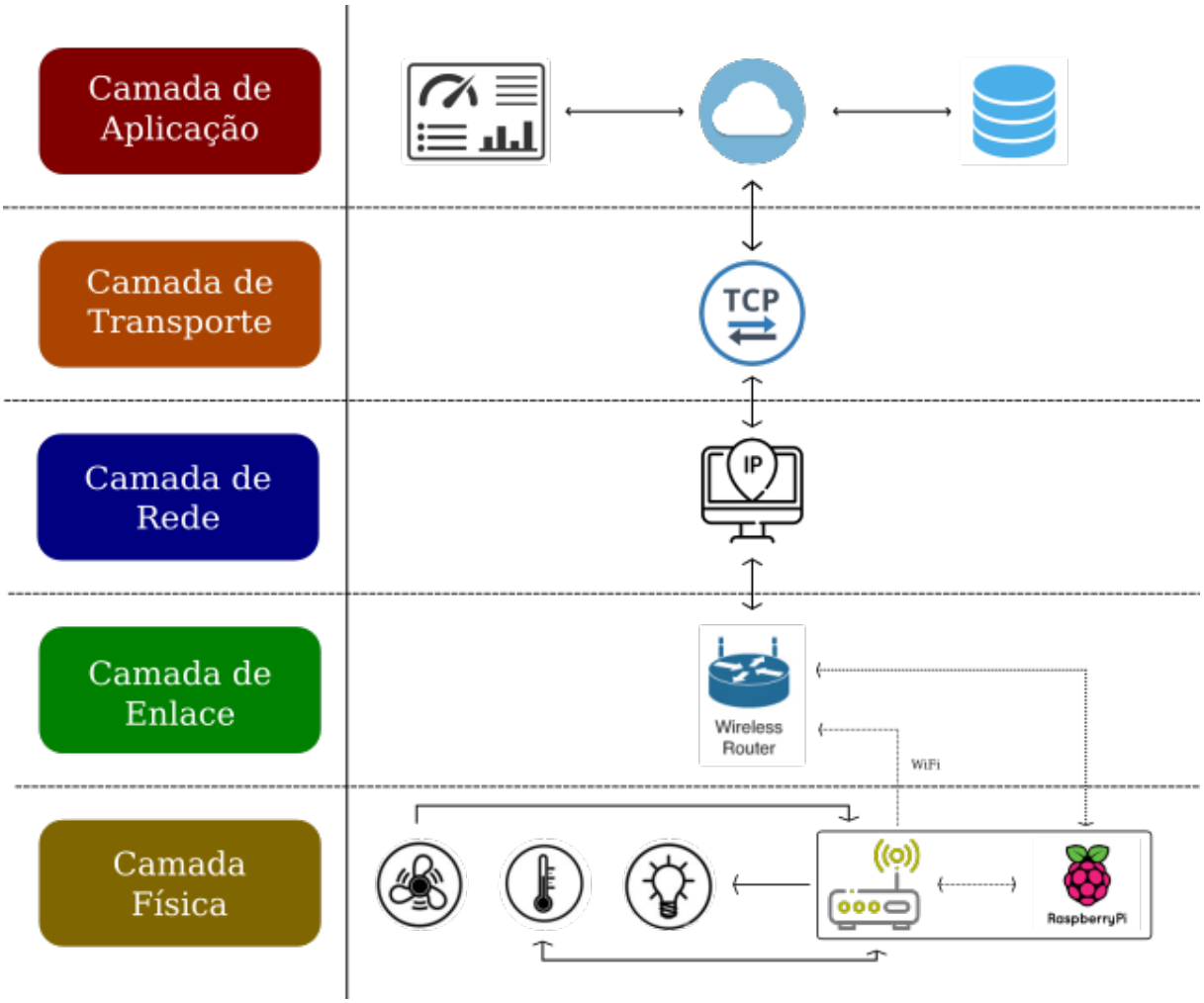

Figura 7 – Proposta da arquitetura para pequeno porte.

A camada física concentra os sensores, atuadores, *gateways* e controladores. Nesse caso, os sensores fornecem dados para os controladores ou dispositivos com capacidade de processamento dos dados. Os atuadores são aqueles que recebem ações dos controladores ou dispositivos com capacidade de processamento. Existem também dispositivos que atuam como

sensores e atuadores, gerando dados e recebendo controles de ações. Isso pode ser visualizado nos três dispositivos na imagem que conversam com os controladores.

Os controladores ou dispositivos com capacidade para processamento são apresentados na Figura 7 pelo *Raspberry PI* ou dispositivos com capacidade de processamento. Essa parte da arquitetura é muito importante para o funcionamento geral de qualquer sistema de IoT, pois se faz necessário uma central de controle dentro de cada ambiente que necessita ser controlado para gerenciar eventualidades como queda de internet ou sobrecarga de chamadas nos servidores de hospedagem. Na proposta da arquitetura de pequeno porte, apresentaram-se linhas pontilhadas; para esse caso, os dois modelos são funcionais. Além disso, existem alguns dispositivos, como, por exemplo, o ESP32, que possui capacidade de processamento de dados e envio de dados para a internet diretamente. O ESP32 não tem a capacidade de processamento de um *Raspberry PI*, por exemplo, porém, consegue ser um microcontrolador que conecta os dispositivos diretamente à internet sem a necessidade de um dispositivo final de alta capacidade.

As camadas de enlace, rede e transporte usam os protocolos TCP/IP para a Internet, mantendo-se esse modelo porque se trata de uma rede doméstica.

A camada de aplicação, por fim, apresenta uma nuvem, um banco de dados e uma *dashboard* de controle. A nuvem computacional seria um conjunto de máquinas espalhadas que se comunicam e conseguem gerenciar as solicitações feitas pelos controladores e dispositivos finais de alto nível. O banco de dados nesse caso pode ser relacional, não relacional ou orientado a cache; o importante aqui é guardar o dado de forma que se tenha um histórico de quando o dado foi gerado e qual dado o sensor coletou, além de outras informações que ajudem a determinar periodicidade ou padrões. Por fim, a *dashboard* é uma plataforma para visualização dos dados gerados e possíveis relatórios com esses dados. Essa *dashboard* pode ser uma solução em software web, *desktop* ou mobile.

Com uma visão geral desta arquitetura de pequeno porte, conseguimos mapear um fluxo de dados geral. Isto é, um caminho padrão por onde o dado trafega na arquitetura e com quais componentes esse dado se comunica. A Tabela 8 apresenta o fluxo de dados para a arquitetura de pequeno porte.

Neste fluxograma, o dado é lido no ambiente. Após isso, o dado é enviado para um dispositivo de controle, que pode ser um microcontrolador, como demonstrado no fluxograma. Este dispositivo pode encaminhar o dado para um dispositivo de processamento local com mais potência ou diretamente para um servidor na nuvem. Caso exista esse dispositivo de processamento local, o microcontrolador envia o dado para ser processado localmente e depois encaminhado para a nuvem. Na nuvem, o dado é tratado novamente de acordo com as especificidades da implementação e é salvo no banco de dados. Paralelamente, o dado é exibido no painel de controle e a aplicação verifica se um evento deve ser executado a partir do valor de entrada desse dado. Caso um evento seja executado, a lógica de programação deve disparar esse evento, que será executado em um dispositivo presente no ambiente.

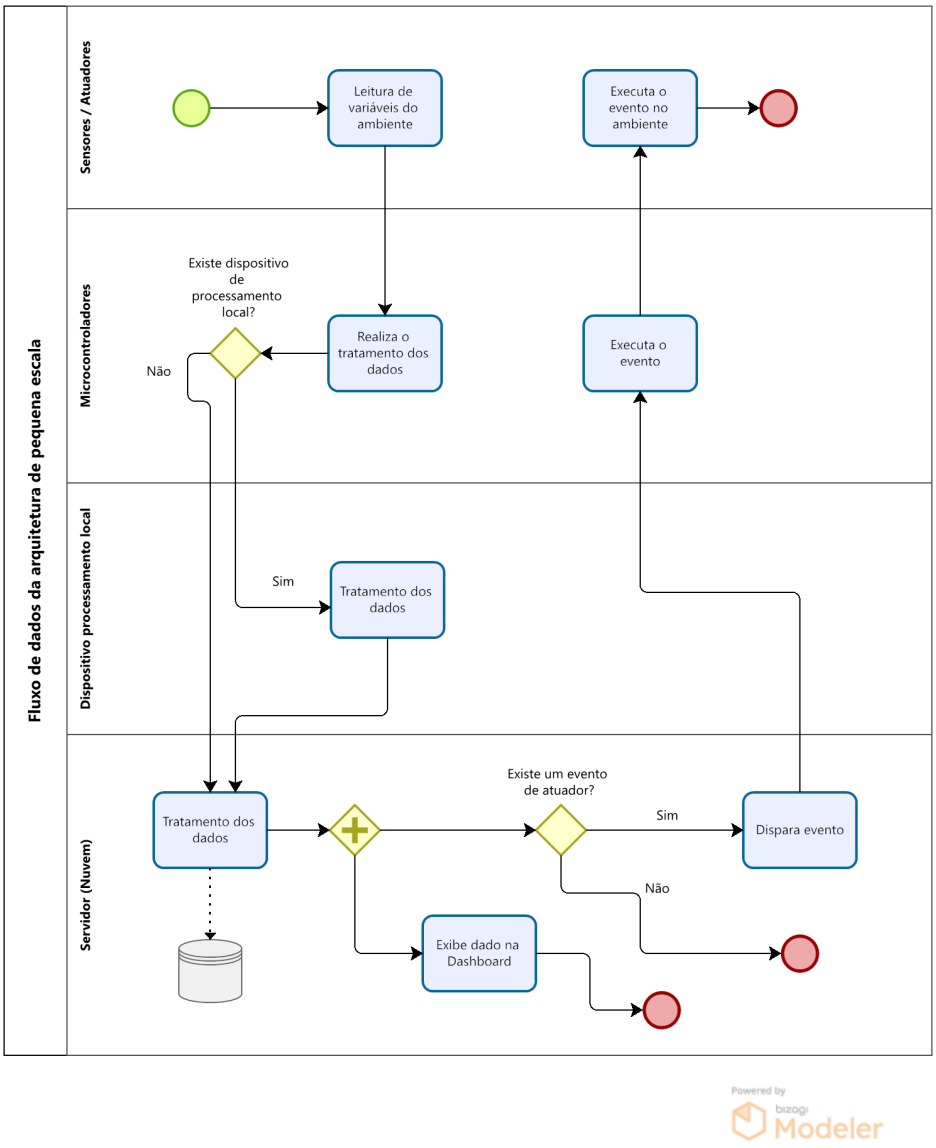

Figura 8 – Fluxo de dados da arquitetura de pequena escala

4.2.2 Média Escala

As arquiteturas de média escala apresentadas no Apêndice C diferem pela repetibilidade arquitetural na camada física, com seus sensores, atuadores e *gateways*. Essa diferença entre a arquitetura de pequena escala e a de média escala ocorre pela necessidade de replicar um conjunto de dispositivos com um controlador por ambiente individual encontrado no ambiente de médio porte. Ou seja, é necessário replicar os dispositivos da camada física de forma que eles possam ser distribuídos em um ambiente maior e consigam controlar e gerenciar esse ambiente. Outra diferença observável está no uso de protocolos para redes privadas, como o *6LowPAN* (IPv6 em Redes de Área Pessoal sem Fio de Baixa Potência), e de comunicação de longa distância, como o *LoRa*.

De forma semelhante às arquiteturas de pequena escala, apresentamos uma arquitetura genérica para as arquiteturas de médio porte, que pode ser vista na Figura 9. Essa arquitetura é, em sua maior parte, baseada no trabalho de Rahmani *et al.* (2018). O primeiro ponto divergente da arquitetura de pequena escala é um componente *6LowPAN*. Essa arquitetura usa o *6LowPAN* como padrão para redes privadas de conexão, que podem ser *ZigBee*, *6LowPAN* ou outras. Essas redes gerenciam sensores e atuadores isolados da Internet, expondo apenas uma interface para acesso. Essa abordagem é utilizada em arquiteturas de médio porte devido à necessidade de isolar alguns componentes do restante da rede, seja por motivos de segurança, isolamento ou manutenção.

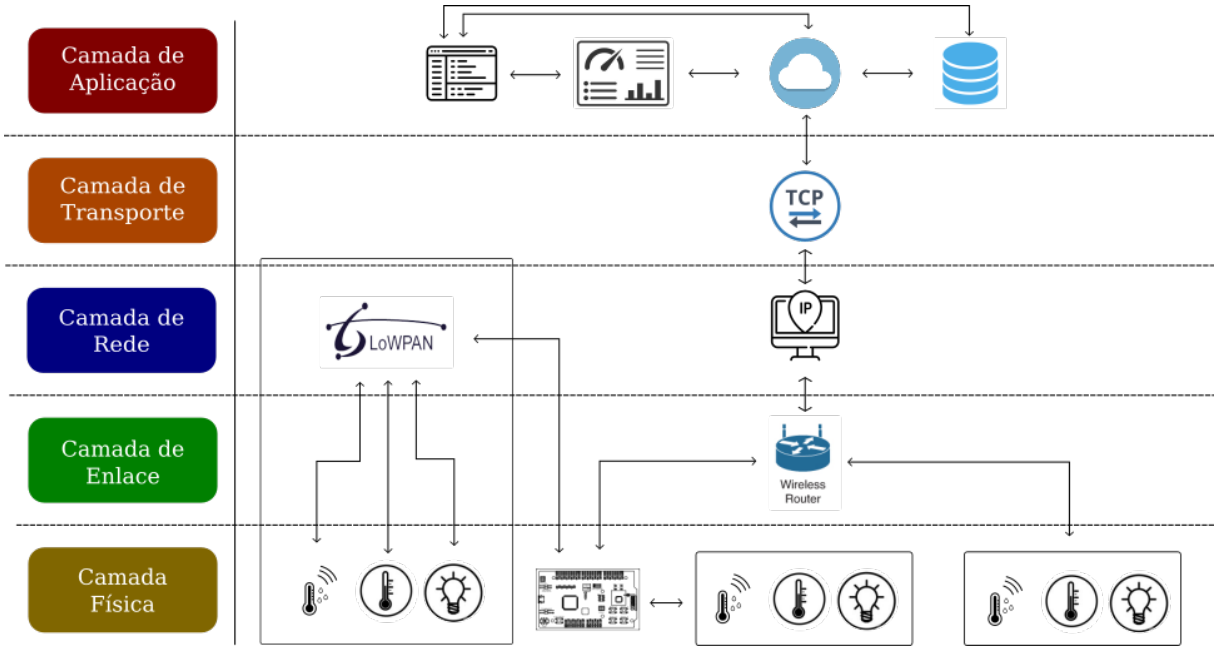

Figura 9 – Proposta da arquitetura para médio porte.

Na camada física, os componentes e estruturas são semelhantes aos ambientes de pequena escala quando tratamos de ambientes pequenos, pois as arquiteturas de média escala são extensões das arquiteturas de pequena escala. Algo interessante a mencionar nesse caso é a replicação dos padrões de controladores, sensores e atuadores em diversos ambientes. A Figura 9 exibe duas estruturas de sensores/atuadores. A primeira está conectada a um dispositivo final com alguma capacidade de processamento. Essa abordagem é interessante quando é necessário realizar o tratamento nos dados antes de enviá-los para uma nuvem computacional. Outra vantagem dessa abordagem está no suporte para operar por um tempo sem Internet, pois o dispositivo consegue manter um histórico dos dados que não foram enviados. A outra situação envolve dispositivos com menos capacidade de processamento ligados aos sensores/atuadores, que se conectam diretamente à Internet. Nesse caso, é necessário que a carga e o processamento da informação sejam realizados na nuvem computacional.

As camadas de enlace, rede e transporte são iguais às camadas das arquiteturas de pequena escala, com a diferença das redes privadas que podem ser utilizadas nesses ambientes

de médio porte.

Na camada de aplicação, existe a adição da análise de dados. Nessa camada, há a necessidade de manter bancos de dados, nuvem computacional e *dashboard*. Porém, existe a necessidade de aumentar a capacidade computacional das máquinas que suportam essas aplicações para que elas continuem escaláveis para ambientes de médio porte. Esse componente de análise de dados pode conter operações de redução nos dados ou até mesmo inteligência artificial.

Um mapeamento do fluxo de dados da arquitetura proposta para média escala é apresentado no fluxograma da Figura 10. Nesse fluxograma, foi adicionada uma nova pista chamada "Microcontroladores em nuvem privada". Essa pista descreve os protocolos de rede privada que podem ser utilizados. Há um aumento da complexidade com a existência de redes públicas e privadas compartilhando o mesmo ambiente de software. Além disso, outra diferença observável está na atividade "Análise dos dados e geração de informação" na pista de Servidor (Nuvem). Essa atividade demonstra que os dados gerados no ambiente, juntamente com outros dados lidos e armazenados, podem gerar novas informações. Essas novas informações podem simular inteligência nesses ambientes.

4.2.3 Larga Escala

Por último, propomos uma arquitetura genérica para aplicações de grande escala que pode ser visualizada na Figura 11. Essa arquitetura é semelhante à proposta por Viswanath *et al.* (2016). Diferentemente da arquitetura proposta por Viswanath *et al.* (2016), as redes privadas para aplicações de grande porte não se limitam às apresentadas pelo autor, devido à necessidade de integrar essas redes privadas que foram utilizadas nas arquiteturas de médio porte. Sua principal diferença em relação às aplicações de médio porte está na necessidade de tratar grandes volumes de dados em tempo real ou com tempo de resposta quase real.

A camada de dispositivos físicos engloba todas as organizações descritas em arquiteturas de pequeno e médio porte, abrangendo as mais diversas formas de redes privadas e comunicação com a Internet. Essa abrangência tem como pressuposto que arquiteturas de grande porte são a integração de diversas arquiteturas e aplicações menores.

A camada de aplicação apresenta a maior diferença em comparação com as arquiteturas menores. Aqui são empregadas tecnologias e aplicativos com foco no processamento do dado em tempo real e na visualização do dado como um fluxo de dados. São utilizadas tecnologias de *Map-Reduce* e aprendizado de máquina para extração de informações e padrões. Além disso, uma aplicação nessa escala deve acessar aplicações externas com frequência. Por fim, é necessário um cuidado especial a esse nível de aplicação com escalabilidade, elasticidade, manutenibilidade, segurança e diversos outros requisitos não funcionais da engenharia de software.

O fluxo de dados para arquiteturas de grande porte é detalhado na Tabela 12. As grandes

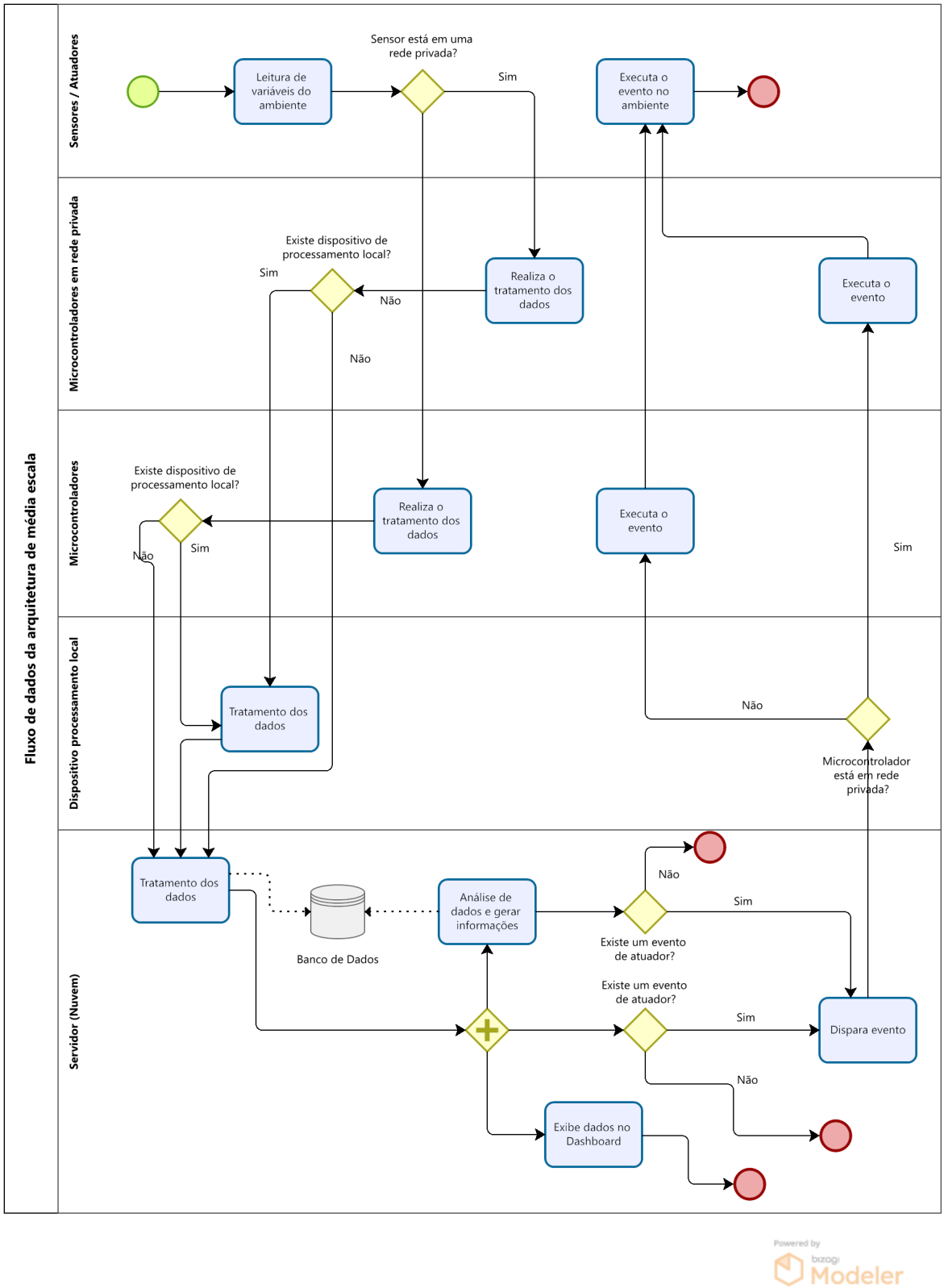

Figura 10 – Fluxo de dados da arquitetura de média escala

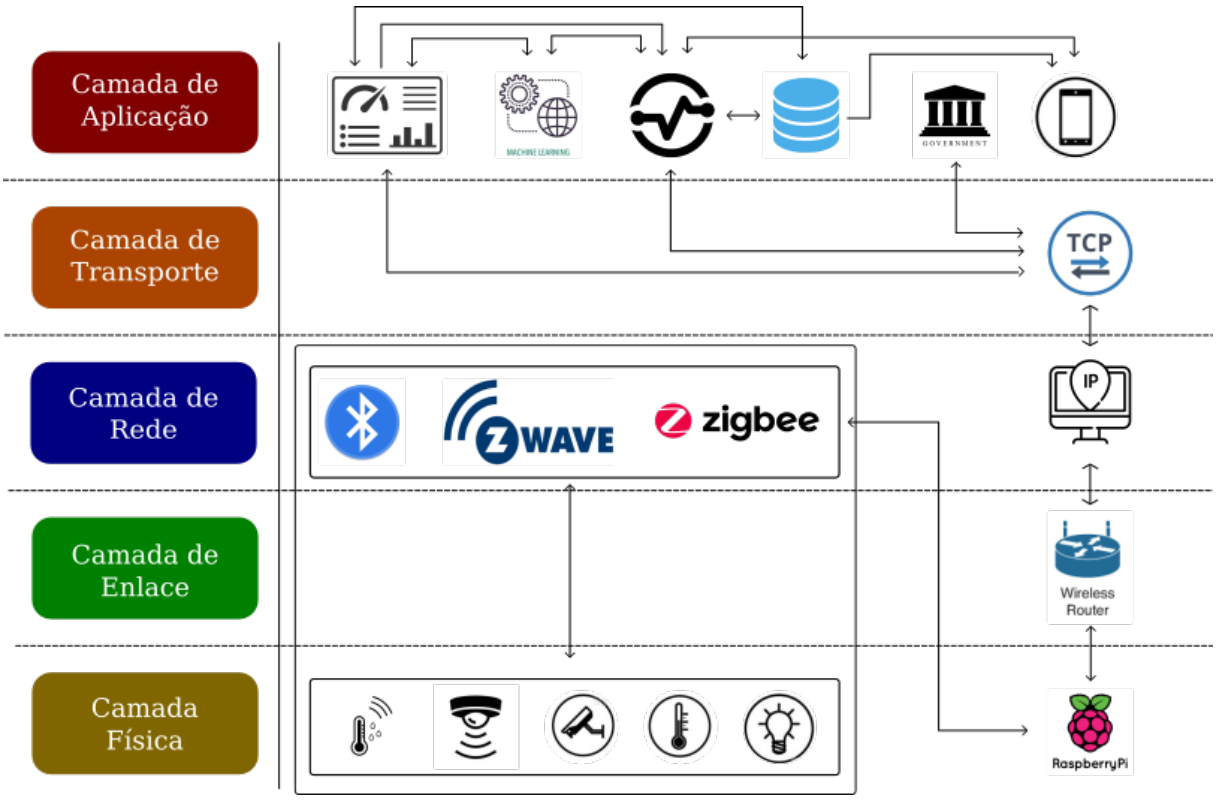

Figura 11 – Proposta da arquitetura para grande porte.

diferenças desse fluxograma de grande porte para o fluxograma de médio porte estão na pista de "Servidor (Nuvem)". Neste fluxograma, temos aplicação de algoritmos de Inteligência Artificial para determinar padrões nos dados coletados e definir eventos de maneira automática. A complexidade das aplicações de grande porte está no tratamento desses *streams* de dados.

4.3 Considerações Finais

Esse capítulo descreveu as arquiteturas genéricas e os fluxogramas mapeados a partir dos exemplos de arquiteturas de IoT encontradas na literatura. Pode-se observar que a complexidade das aplicações de IoT aumenta conforme a escala da aplicação aumenta. As menores aplicações têm poucos recursos de hardware e software, enquanto as grandes aplicações precisam de grandes quantidades de recursos computacionais para processar os dados.

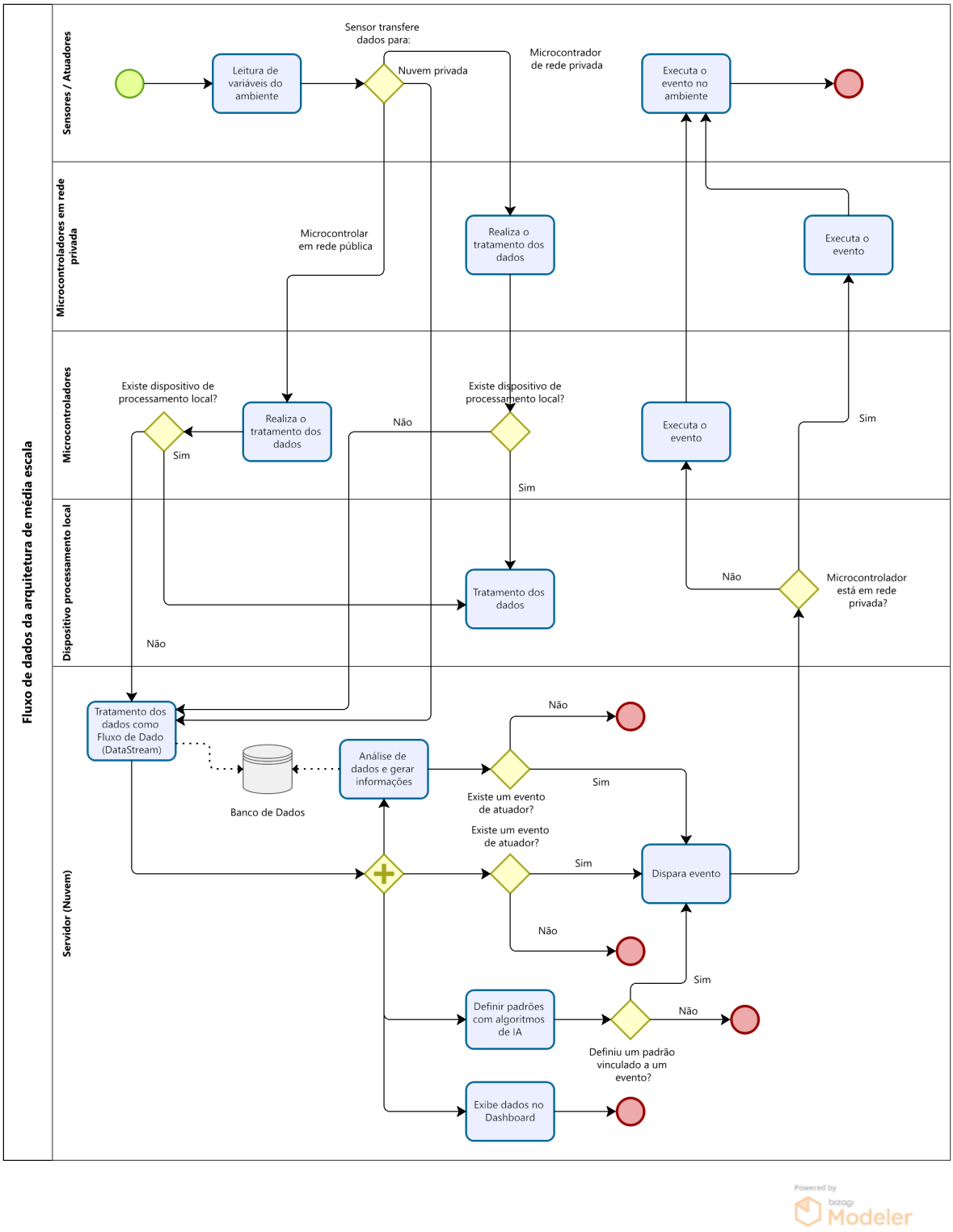

Figura 12 – Fluxo de dados da arquitetura de grande escala

CAPÍTULO

5

PROTOTIPAÇÃO DO MODELO ARQUITETURAL

5.1 Considerações Iniciais

Com os modelos de arquiteturas definidos e os fluxogramas mapeados, a próxima etapa do projeto é desenvolver uma aplicação funcional com base na arquitetura de médio porte capaz de atender à demanda para o controle dos ares-condicionados do ICMC. Esse modelo foi desenvolvido e testado no LaSDPC.

Na Seção 5.2, apresentamos uma descrição detalhada desse modelo arquitetônico. Apresentando na Seção 5.3 os componentes físicos utilizados e as bibliotecas para o controle desses dispositivos físicos. A Seção 5.4 tem o objetivo de apresentar as telas do *front-end*, que exibem os dados medidos pelos sensores e permitem a interação com os atuadores. A Seção 5.5 apresenta a lógica da aplicação contida no *back-end*, a qual contém duas aplicações que são utilizadas para o controle dos usuários e dispositivos, duas *Bridges* que são responsáveis por gerenciar a transmissão de dados entre o MQTT e o Apache Kafka e, por fim, o acesso aos dois bancos de dados utilizados por essa aplicação. Por fim, na Seção 5.6, apresentamos algumas considerações sobre o modelo arquitetural desenvolvido.

5.2 Descrição do Protótipo

Componentes, protocolos, arquiteturas e *frameworks* foram empregados na criação do protótipo. Essa representação pode ser subdividida em quatro grupos distintos: Controle dos Dispositivos Físicos, *Bridges* e Kafka, Gerenciamento da Aplicação e Bancos de Dados. Cada um desses grupos assume a responsabilidade pela coordenação de um conjunto específico de funções dentro da arquitetura. Nas próximas seções, vamos fornecer descrições detalhadas e elucidativas desses componentes. Essa divisão é melhor apresentada na Figura 13.

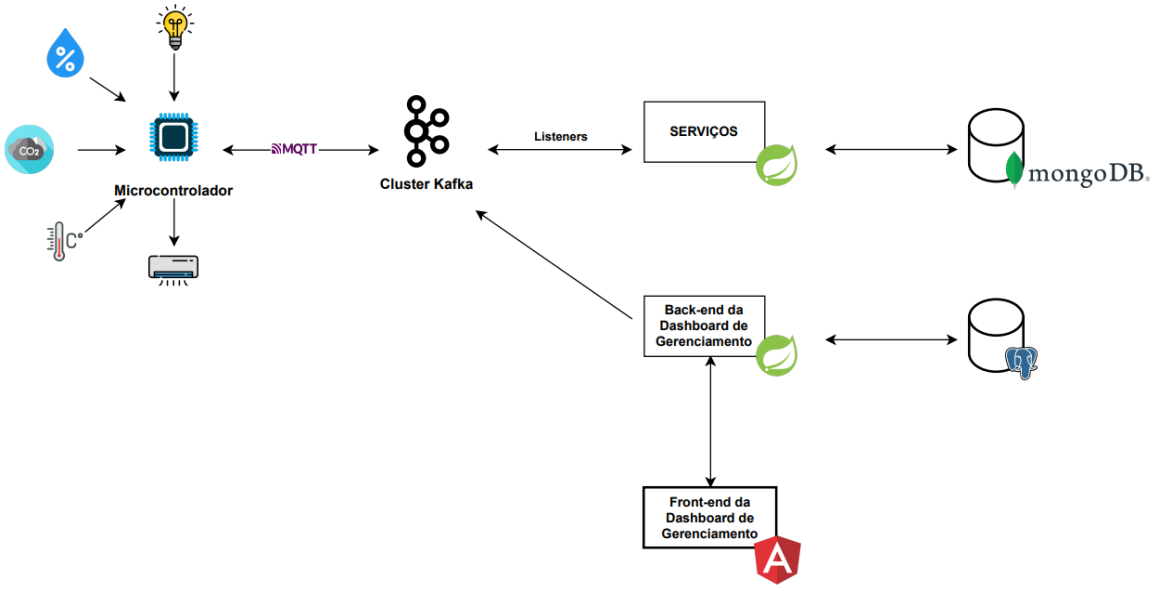

Figura 13 – Arquitetura de Médio Porte

A Figura 13 apresenta uma visão geral das interconexões entre os componentes da arquitetura. O microcontrolador está comunicando com os dispositivos físicos, como os sensores de umidade, iluminação, CO2 e temperatura, e com o atuador do ar-condicionado. Essa comunicação é feita por protocolos como o I2C.

O microcontrolador também necessita se comunicar com os servidores, comunicação que é feita por meio do protocolo MQTT. Entre a comunicação do MQTT com o Kafka, há duas *Bridges*, códigos escritos em Java que lêem de um tópico do MQTT e publicam em um tópico do Apache Kafka e vice-versa. A principal função das *Bridges* é converter os dados entre os protocolos MQTT e o Apache Kafka.

Ainda na representação da arquitetura, nos deparamos com blocos que representam os bancos de dados. Os dois bancos de dados utilizados para o desenvolvimento do protótipo foram o MongoDB e o PostgreSQL. Por fim, a arquitetura apresenta as aplicações escritas em Java que realizam o controle da aplicação e a aplicação "Front-end da Dashboard de Gerenciamento", que é utilizada para visualização dos dados e solicitação de ações no ambiente. Esta arquitetura é descrita com mais detalhes nas próximas seções.

5.3 Hardware e Dispositivos

Vários dispositivos foram utilizados para a implementação dessa arquitetura. Esses dispositivos vão desde os microcontroladores, sensores e atuadores no ambiente até a infraestrutura necessária para subir a aplicação no LaSDPC.

A infraestrutura tem foco primário no controle dos ares-condicionados. Foram empre-

gados ESP32, que se conectam aos sensores de tensão ACS712 de 30A. Esses sensores são usados para medir correntes AC/DC do ar condicionado, e atuadores de infra-vermelho, que são empregados para realizar operações de ligar/desligar o ar condicionado, além de alterar a temperatura e o modo de funcionamento do ar. A Figura 14 apresenta as mudanças necessárias no ambiente para a instalação desses dispositivos.

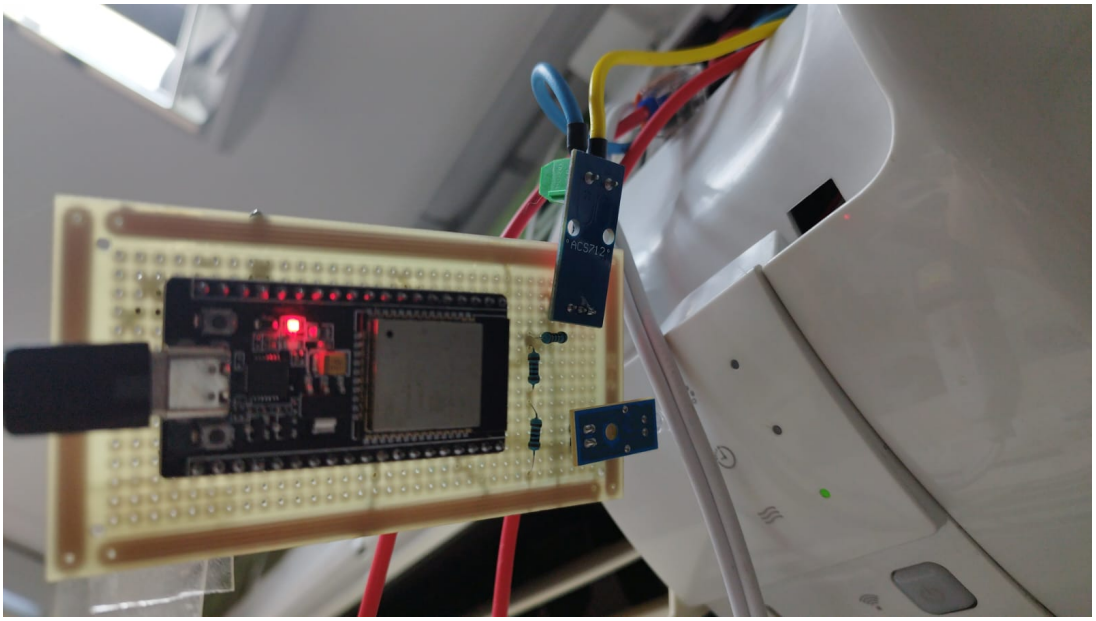

Figura 14 – Ar condicionado ajustado no ambiente físico para a arquitetura

Na Figura 14 observamos um ESP32 com uma *protoboard*, na Figura 15 (1) demonstramos esse componente com uma placa de expansão, a *protoboard* e a placa de extensão são utilizadas para estender os pinos do ESP32 e auxiliar nas conexões com outros dispositivos. No canto inferior direito da Figura 14 está localizado o atuador infravermelho, que é demonstrado em melhor detalhe na Figura 15 (2). O sensores de tensão ACS712 é apresentado no canto superior direito da Na Figura 14, este componente pode ser melhor observado na Figura 15 (3). O sensor de tensão ACS712 realiza uma leitura da corrente elétrica e determina se o ar condicionado está ligado ou desligado. Esse dado é coletado e enviado para o broker MQTT. A partir desse dado, conseguimos determinar qual configuração pode ser executada no ambiente a partir da interface do *Front-end*. O infravermelho funciona como um controle remoto e envia comandos para o ar condicionado.

5.3.1 Biblioteca de Controle de Fluxo de Laboratório

Com o propósito de administrar esses dispositivos, uma biblioteca em C++ para Arduino IDE foi criada. Essa biblioteca recebeu o nome de Biblioteca de Controle de Fluxo de Laboratório (BibFluxLab). O objetivo principal da BibFluxLab é padronizar a publicação de dados provenientes dos dispositivos ESP, unificando a estrutura de dados utilizada para essa finalidade.

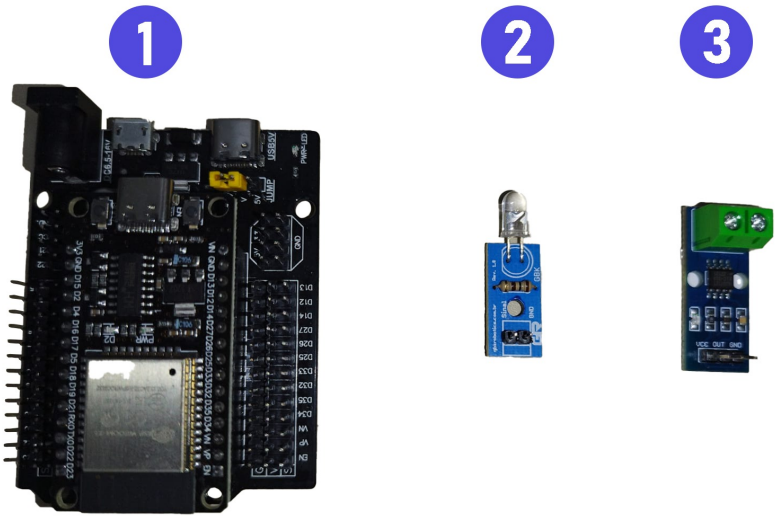

Figura 15 – Dispositivos utilizados no ambiente do laboratório

Além disso, a BibFluxLab disponibiliza interfaces que simplificam a realização de operações de subscrição.

A configuração da BibFluxLab necessita de definições de acesso à rede local (Wi-Fi) e do identificador único do dispositivo cadastrado no Dashboard de Gerenciamento. O identificador único é utilizado com o objetivo de identificar qual o microcontrolador específico que está gerando dados ou necessita executar ações, funcionando de forma semelhante aos endereços postais para a entrega de mensagens pelos sistemas de correios.

Adicionalmente, a BibFluxLab oferece métodos para a publicação de dados de temperatura, umidade, níveis de CO2, detecção de movimento e intensidade de iluminação. Além disso, ela possibilita a criação de uma classe que herda funcionalidades para executar ações como ligar/desligar, ajustar a temperatura, alterar o modo de operação do ar-condicionado, bem como um método para controlar o estado das lâmpadas.

Por fim, a BibFluxLab realiza a comunicação entre os dispositivos ESP e o broker MQTT por meio de uma instância do Eclipse Mosquitto instalado no LaSDPC. Essa instância do Eclipse Mosquitto funciona como um intermediário entre a transmissão dos dados do microcontrolador e o Apache Kafka.

5.3.2 Nuvem Computacional

Na nuvem computacional do LaSDPC, foi configurado todo o aparato de infraestrutura e sistemas necessários para a execução do protótipo. A nuvem do LaSDPC é composta por diversos *hosts* com seus recursos divididos entre Máquinas Virtuais (VMs). Cada *host* é configurado com as características apresentadas na Tabela 1.

Tabela 1 – Configurações dos Hosts Físicos

-	**Configurações**
Sistema Operacional	Ubuntu 22.04 LTS
Processador	AMD FX 6300
Memória	32GB de RAM DDR3
Discos	1 disco de 250GB com o SO 1 disco NVMe 1TB para vms 1 disco sata 3 - 500GB para imagens dos containers 1 disco de 2TB para backup

Para a execução do protótipo, são utilizadas duas VMs. A VM01 foi empregada para a instalação dos softwares terceiros necessários para o funcionamento da aplicação, e a VM02 foi empregada para a instalação das aplicações de *Front-end*, *Back-end* e *Bridges*. Essas máquinas virtuais apresentam as seguintes configurações: Ubuntu 22.04 LTS, 8GB de memória RAM, 1 disco virtual de 100GB com o SO e 1 disco virtual de 50GB para as imagens do *Docker*.

Os recursos instalados nessas VMs são gerenciados pelo *Docker*. O *Docker* permite a instalação de aplicações sem a necessidade de instalar bibliotecas e softwares diretamente nessas VMs. Em outras palavras, a aplicação executa em *containers* do *Docker*, os quais gerenciam as necessidades de cada aplicação. A Tabela 2 apresenta as aplicações e a quantidade de instâncias do *Docker* em execução para cada VM. É possível observar que a VM01 executa aplicações terceiras e a VM02 executa aplicações desenvolvidas para o funcionamento lógico do modelo arquitetural.

Tabela 2 – Distribuição de Aplicações para as VMs

Máquinas Virtuais	**Aplicações**	**Qtde. de Instâncias *Docker***
VM01	Eclipse Mosquitto 1.5.5 Zookeeper Apache Kafka PostgreSQL 14.2 MongoDB 4.0.5	1 2 4 1 1
VM02	*Front-end* *Back-end*: “Núcleo de Gerenciamento” *Back-end*: “Serviço de Controle de Tópicos” *Bridge*: MqttKafBridge *Bridge*: KafMqttBridge	1 1 1 1 1

As aplicações que estão em execução na VM02 serão melhor explicadas nas próximas seções. Essas aplicações foram criadas exclusivamente para o gerenciamento do protótipo.

5.4 Front-end

Desenvolvido em *Angular*, o *Front-end* da aplicação é responsável pelo gerenciamento de ambientes, gerenciamento dos dispositivos cadastrados, listagem dos dados coletados pelos sensores e acionamento para a execução de ações nos atuadores.

O gerenciamento de ambiente é o nome dado ao cadastro de instituições, blocos, pisos e salas. Esses cadastros são fundamentais para determinar onde o dispositivo está instalado no ambiente físico. Isso facilita a rastreabilidade do dispositivo e auxilia na identificação física de dispositivos que possam estar com problemas.

O gerenciamento dos dispositivos cadastrados está vinculado ao cadastro de microcontroladores, sensores e atuadores. Os microcontroladores obrigatoriamente devem estar ligados a uma sala; o registro desse dado é fundamental para determinar em qual instituição, bloco, piso e sala o microcontrolador está instalado. Um microcontrolador está vinculado a sensores e atuadores e no cadastro de microcontrolador são determinados os tipos de atuadores e sensores e quais são os seus tipos. Esses registros são feitos com o intuito de rastrear de onde os dados coletados provêm e qual microcontrolador em específico precisa executar uma ação no ambiente. Como identificador único, cada microcontrolador, sensor e atuador possui um Identificador Único Universal (UUID). Esses UUIDs são usados pela BibFluxLab para definir quais dispositivos o ESP32 está controlando.

Um exemplo da tela de listagem de microcontroladores pode ser visualizado na Figura 16. Destacam-se os filtros de instituição, bloco, piso e sala selecionados e os microcontroladores desse ambiente listados.

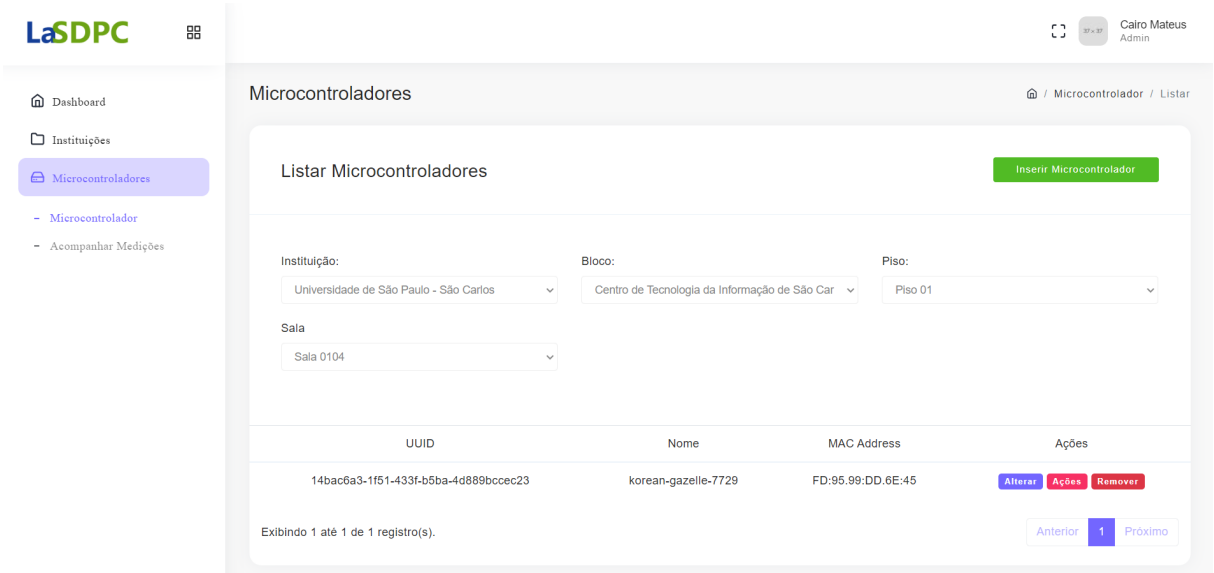

Figura 16 – Tela de listagem de microcontroladores

A listagem dos dados coletados pelos sensores fica em um *menu* chamado de “Acompanhar Medições”, nesta página há um filtro para selecionar o ambiente e a listagem correspondente

aos sensores localizados nesse ambiente. Ao clicar em um botão chamado “Medições”, um *modal* abre e apresenta a lista com as medições daquele dispositivo.

Por fim, como pode ser visto na Figura 16, existe um botão de ações dentro da listagem de microcontroladores. Esse botão de ações abre um *modal* que lista todos os atuadores cadastrados para o microcontrolador, com as opções de ações que podem ser executadas dentre os tipos de atuadores presentes nesse microcontrolador.

5.5 Back-end

O *Back-end* do protótipo é composto por dois serviços, duas *Bridges* e as conexões com os bancos de dados (Figura 13). Esses serviços são responsáveis por gerenciar as filas do Apache Kafka, salvar as informações nos bancos de dados, expor *endpoints*, expor as informações coletadas pelos sensores e oferecer um ponto de entrada para a execução de ações no ambiente. Esses serviços foram nomeados de “Núcleo de Gerenciamento” e “Serviço de Controle de Tópicos” para uma melhor divisão e explicação desses componentes.

O “Núcleo de Gerenciamento” é responsável por armazenar os dados de identificação dos microcontroladores, sensores e atuadores. Ele também envia os identificadores únicos dos dispositivos para o Apache Kafka, que são posteriormente recuperados pelo "Serviço de Controle de Tópicos"e registrados no MongoDB para controle e filtro das informações geradas pelo microcontrolador. O “Núcleo de Gerenciamento” também é responsável pela comunicação com o banco de dados PostgreSQL.

Já o “Serviço de Controle de Tópicos” desempenha um papel fundamental no gerenciamento dos produtores e consumidores do Apache Kafka. É aqui que são realizados os cadastros e a recuperação dos dados dos sensores que foram gravados no MongoDB.

Concluímos que esses dois serviços definem o núcleo da aplicação. Nestes serviços temos a lógica de programação para salvar, recuperar e operar as publicações e leituras do Apache Kafka. Nas subseções abaixo descrevemos um pouco sobre os outros componentes do *Back-end*.

5.5.1 Bridges

As aplicações *Bridges* são responsáveis por conectar o Broker MQTT com os tópicos do Apache Kafka, elas servem como conversores de mensagens entre esses dois sistemas de mensageria. O Apache Kafka é utilizado para orquestrar o fluxo de dados na arquitetura, de forma que todos os dados e ações que ocorrem nos dispositivos passam por um tópico do Kafka.

Na definição da arquitetura do Apache Kafka, adotamos a configuração com duas instâncias do Zookeeper e quatro instâncias do Apache Kafka. Esta abordagem foi escolhida com o objetivo de garantir a replicabilidade dos nós do Kafka e do Zookeeper. Isso significa que, em caso de falha em alguma das instâncias, a aplicação continuará operando sem interrupções. A

Figura 17 apresenta essa arquitetura. Observamos nela a existência de produtores, em verde, que produzem dados nos tópicos do Apache Kafka. Um *cluster* do Apache Kafka com 4 instâncias, em azul, criam redundância e gerenciam as publicações nos tópicos. O Zookeeper, em vermelho, é utilizado para a configuração dos tópicos, lista de controle de acessos e associação dos *clusters* do Apache Kafka. E por fim, em roxo, os consumidores realizam a leitura das mensagens publicadas nos tópicos pelos produtores.

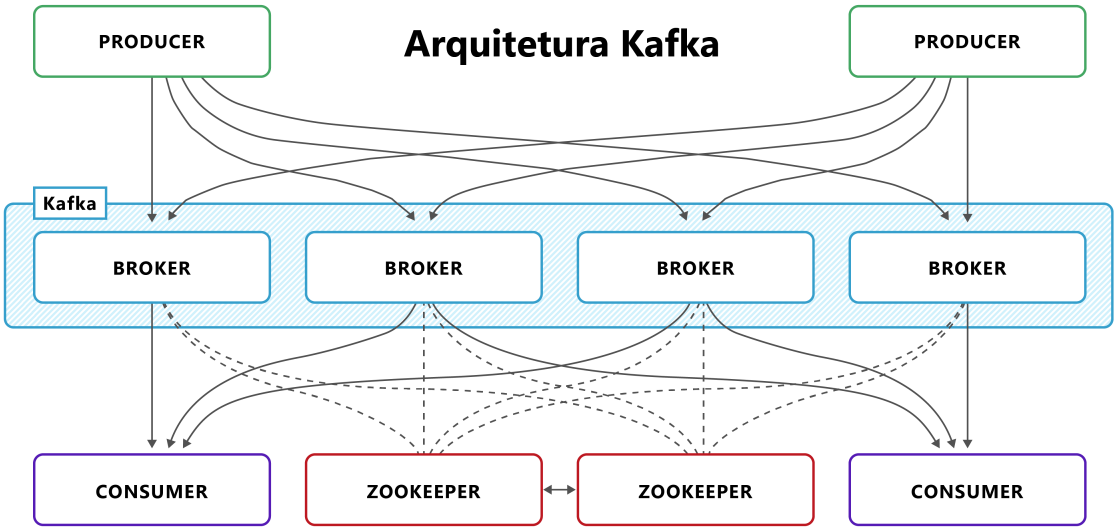

Figura 17 – Arquitetura do Kafka

A arquitetura apresenta duas *Bridges*. A primeira bridge, nomeada de MqttKafBridge, é responsável por interceptar as mensagens que são publicadas em um tópico do MQTT e realizar a conversão desse dado para o tópico correspondente do Apache Kafka. A segunda bridge, KafMqttBridge, é responsável por interceptar os dados dos tópicos do Apache Kafka e enviá-los para o tópico do MQTT. Uma descrição visual desse processo é apresentada na Figura 18.

Como ilustrado na Figura 18, as *Bridges* foram desenvolvidas com a finalidade de efetuar a conversão de dados entre os ambientes MQTT e Kafka. Este processo ocorre de maneira eficiente, com os dados transmitidos no formato JSON, e a vantagem adicional de que os tópicos do Kafka e MQTT não precisam necessariamente coincidir. Normalmente, o identificador único do dispositivo, um UUID neste caso, é transmitido dentro do corpo da requisição. Porém, existe um cenário entre o KafMqttBridge e o MQTT em que o UUID é transferido no tópico, isso ocorre porque a ação é realizada em um atuador específico. A adição do UUID no tópico permite a entrega da mensagem diretamente para o ESP32 no qual o atuador está registrado. Essa abordagem foi utilizada para melhorar a forma como as entregas são realizadas para os atuadores, a fim de não sobrecarregar os dispositivos físicos.

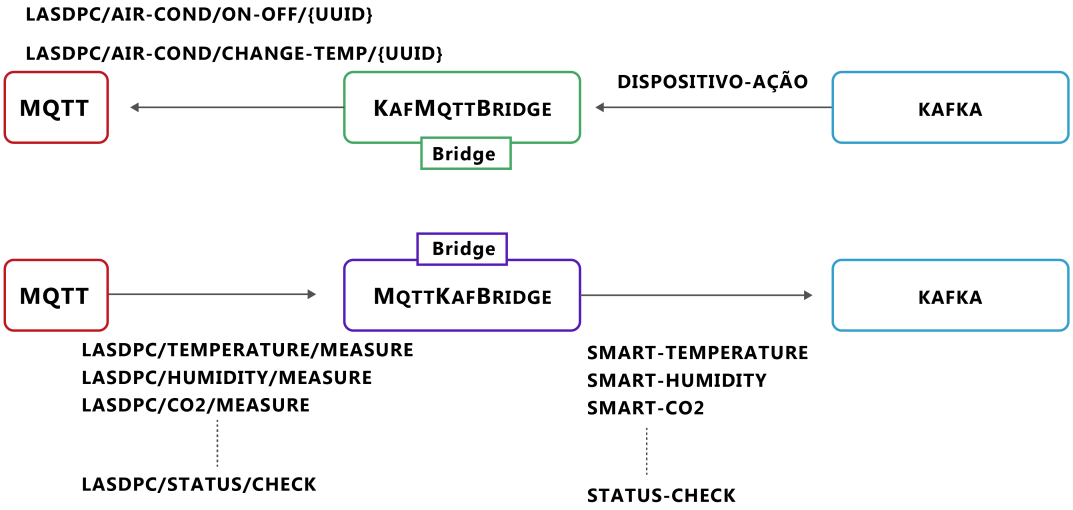

Figura 18 – Descrição visual das *Bridges*

Por fim, concluímos que as *Bridges* são fundamentais para a comunicação entre o protocolo de comunicação MQTT e a plataforma do Apache Kafka.

5.5.2 Banco de Dados

Para o armazenamento de dados no sistema, são utilizados dois bancos de dados: o PostgreSQL e o MongoDB. O PostgreSQL é um banco de dados relacional utilizado por sua capacidade de integridade transacional, capacidade de realizar consultas complexas e robustez. Neste protótipo, ele é utilizado para guardar os dados de cadastro de microcontroladores, sensores e atuadores.

O MongoDB é um banco de dados flexível e escalável, sendo empregado no gerenciamento de dados não estruturados ou semi-estruturados. Sua utilização tem como objetivo principal guardar os dados coletados pelos sensores, pois essas informações podem evoluir à medida que novas necessidades e tratamentos de dados surgem. Além disso, ele fornece funções para aplicar map-reduce nos dados, ampliando ainda mais a sua utilidade na análise e extração de padrões a partir das informações armazenadas.

Assim, cada banco de dados é utilizado para aproveitar suas melhores características. O “Gerenciamento da Aplicação” é responsável pelas conexões e operações com os bancos de dados.

5.6 Considerações Finais

Este tópico apresentou o protótipo do modelo arquitetural. Descrevemos a arquitetura com os dispositivos físicos, a biblioteca desenvolvida para o gerenciamento desses dispositivos de borda computacional, a infraestrutura que oferece suporte para o projeto, o *Front-end*, o controle da aplicação no *Back-end*, as *Bridges* que conectam o MQTT e o Apache Kafka e, por fim, uma descrição dos bancos de dados.

Demonstramos a diversidade de tecnologias empregadas neste protótipo do modelo arquitetural, bem como expressamos a complexidade no desenvolvimento deste tipo de aplicação.

CAPÍTULO

6

RESULTADOS

6.1 Considerações Iniciais

O objetivo deste capítulo é apresentar um estudo experimental inicial com o protótipo do modelo. Vamos medir o tempo de resposta das requisições HTTP e a capacidade de leitura e escrita de ponta a ponta no modelo arquitetural. Para este estudo vamos focar na geração de carga de trabalho considerando duas distribuições de probabilidade: uniforme e exponencial.

Uma publicação de ponta a ponta refere-se a uma publicação que começa no *Front-end*, atravessa o "Núcleo de Gerenciamento", chega ao Apache Kafka e é convertida para a *Bridge* KafMqttBridge. O dado é então publicado no MQTT e chega ao ESP32. A Figura 19 apresenta uma descrição desta publicação de ponta a ponta.

Figura 19 – Ações de ponta-a-ponta do modelo arquitetura proposto

A realização desses experimentos utilizou o modelo arquitetural e um dispositivo ESP32, que é responsável por ler os dados e executar uma ação no ambiente. Para verificar se a ação foi executada, publicamos em um tópico do MQTT chamado *"lasdpc/status/check"*. Realizamos uma subscrição nesse tópico MQTT e verificamos a quantidade dessas que são lidas corretamente depois de passar por todo o processo descrito na Figura 19.

Para os experimentos, utilizamos a ferramenta K6 (K6, 2023). O K6 é uma ferramenta de código aberto e serviço em nuvem que facilita o teste de carga para desenvolvedores e engenheiros de controle de qualidade, fornecendo vários *plugins* que podem ser usados para

estender as funções básicas da ferramenta. Utilizamos uma biblioteca do K6 chamada “k6/http” para realizar os experimentos com requisições do tipo HTTP. Além disso, usamos o *plugin* “k6-mqtt” para a leitura das mensagens que chegam no tópico *“lasdpc/status/check*” do MQTT.

O K6 também oferece diversos cenários de testes, como interações compartilhadas, interações por usuários virtuais, usuários virtuais constantes, variação em usuários virtuais, taxas de requisições constantes e taxas de requisições crescentes. Neste trabalho, utilizamos taxas de requisições constantes para realizar testes de requisições constantes e taxas de requisições crescentes para validar como o modelo arquitetural se comporta com a elevação na quantidade de requisições de maneira exponencial.

Para os experimentos, foram utilizadas 10 máquinas virtuais localizadas no LaSDPC. Cada uma dessas máquinas está configurada com uma CPU de 4 núcleos e 16 GB de memória RAM. Os comandos do K6 foram colocados dentro de um arquivo *batch* capaz de executar as rotinas do experimento. A função "Crontab"do Ubuntu foi utilizada para agendar e realizar a execução desse *batch*. Dessa forma, as rotinas executaram ao mesmo tempo e criaram a carga necessária para os testes. As Seções 6.2 e 6.3 descrevem melhor os experimentos realizados.

6.2 Descrição do 1º Experimento

O 1º experimento tem o objetivo de medir o comportamento do protótipo para requisições constantes ao longo do tempo. Nele, propomos análises de requisições HTTP sequenciais. Criamos cinco cenários para o experimento, os quais são melhor descritos na Tabela 3.

Tabela 3 – Entradas para o 1º Experimento

Cenário	**Nº Requisições**	**Mapeamento**
1	100	10 requisições por VM
2	1.000	100 requisições por VM
3	10.000	1.000 requisições por VM
4	50.000	5.000 requisições por VM
5	100.000	10.000 requisições por VM

Em cada segundo são realizadas 4 requisições por cada VM. Dessa forma, o tempo em segundos (***T(s)***) de cada teste é definido pelo número total de requisições dividido pela quantidade de máquinas virtuais, dividido por 4. Esse cálculo é demonstrado na Equação 6.1. Sendo ***R*** o número de requisições e ***V*** o número de máquinas virtuais.

$$T(s) = \frac{R}{V} \div 4 \tag{6.1}$$

A saída para esses experimentos são a média, a mediana, o tempo mínimo das requisições, o tempo máximo das requisições e a porcentagem de publicações do ESP32 para o MQTT que

foram lidas. Utilizamos esses parâmetros para apresentar os resultados dos experimentos na Seção 6.4.

6.3 Descrição do 2º Experimento

O 2º experimento considera a geração de carga de trabalho com base em uma distribuição exponencial. Neste experimento definimos como o protótipo reage a mudanças crescentes no número de requisições. Definimos o uso da equação exponencial 2^x para o experimento, no qual o valor de x começa em 0 e vai sendo incrementado a cada 20 segundos. A Tabela 4 apresenta os 5 cenários de testes propostos para o 2º experimento.

Tabela 4 – Entradas para o 2º Experimento

Cenário	Nº Requisições por VM	Nº Total de Requisições
1	$\sum_{x=0}^{3} 2^x$	150
2	$\sum_{x=0}^{5} 2^x$	630
3	$\sum_{x=0}^{7} 2^x$	2550
4	$\sum_{x=0}^{9} 2^x$	10230
5	$\sum_{x=0}^{11} 2^x$	40950

O *Número de Requisições por VM* é definido pelo somatório da função 2^x começando com o limite inferior de zero. Definimos o limite superior para o cenário 1 como 3 e para cada cenário subsequente aumentamos esse limite em 2. Dessa forma, temos uma progressão linear do limite superior para uma função exponencial.

É importante observar que o *Nº Total de Requisições* é igual ao somatório do *Número de Requisições por VM* multiplicado pela quantidade de máquinas virtuais que existem no servidor (neste cenário, 10 máquinas virtuais). Desta forma, para o cenário 1, temos uma resposta de 15 para o *Número de Requisições por VM* que é multiplicado por 10 para a resposta do *Nº Total de Requisições*.

O número de requisições por segundo aumenta a cada ciclo do somatório. Dessa forma, por exemplo, no cenário 5, para cada máquina virtual temos inicialmente uma taxa de 0,05 requisições por segundo e encerramos com aproximadamente 102 requisições por segundo.

Realizamos este experimento com o intuito de verificar como o modelo arquitetural proposto reage ao aumento crescente das requisições. Por isso, usamos um tempo definido para cada ciclo de requisições, obrigando o modelo arquitetural a lidar com aumentos de requisições repentinos a cada ciclo.

6.4 Resultados

Nesta seção, apresentamos os resultados propostos para os experimentos das Seções 6.2 e 6.3. As Tabelas 5 e 6 mostram os resultados de mínimo, máximo, média, desvio padrão e intervalo de confiança para os experimentos. Os valores mínimos, máximos e médios são obtidos a partir do *Nº Total de Requisições* realizadas. O desvio padrão é obtido a partir das médias dos resultados dos 10 testes realizados nas 10 VMs. O intervalo de confiança utiliza a média, o desvio padrão e um nível de confiança de 95%.

Tabela 5 – Resultados do 1º Experimento

Cenário	Mínimo	Máximo	Média	Desvio Padrão	Intervalo de Confiança
1	2,0629ms	4,8539ms	3,2733ms	0,6758ms	1,9487ms - 4,5979ms
2	1,8958ms	7,8895ms	3,2149ms	0,6723ms	2,7144ms - 3,7153ms
3	1,6214ms	13,0475ms	3,1718ms	0,0945ms	2,9867ms - 3,357ms
4	1,5177ms	19,9358ms	3,1395ms	0,0849ms	2,9732ms - 3,3059ms
5	1,4842ms	43,3394ms	3,1488ms	0,09ms	2,9723ms - 3,3252ms

Na Tabela 5, apresentamos os resultados para o 1º Experimento. É possível observar que o tempo mínimo diminui conforme o número de requisições aumenta. Esse efeito pode acontecer por diversos motivos, como a quantidade de requisições efetuadas gerando um tempo mínimo baixo ou a existência de cache na aplicação. O valor máximo aumenta conforme a quantidade de requisições aumenta. Uma possível causa desse efeito está na sobrecarga causada pelas requisições HTTP. A média se mantém constante nos resultados, indicando que a média está entre 3,1s e 3,3s. O desvio padrão diminui conforme a quantidade de requisições aumenta. Dessa forma, é possível afirmar que os dados possuem menor distorção conforme mais requisições são realizadas. Por fim, temos 95% de confiança de que o valor das requisições esteja entre o valor mínimo e máximo do intervalo de confiança.

Tabela 6 – Resultados do 2º Experimento

Cenário	Mínimo	Máximo	Média	Desvio Padrão	Intervalo de Confiança
1	2,3517ms	4,7103ms	3,2711ms	0,114ms	3,0476ms - 3,4946ms
2	2,2339ms	5,1783ms	3,2599ms	0,098ms	3.0677ms - 3.452ms
3	1,7614ms	8,181ms	3,1030ms	0,08ms	2,9463ms - 3,2598ms
4	1,5657ms	11,1671ms	2,9672ms	0,077ms	2,8164ms - 3,1181ms
5	1,5587ms	18,1358ms	2,984ms	0,07ms	2,8474ms - 3,1205ms

A Tabela 6 apresenta os resultados para o 2º Experimento. Semelhante ao 1º Experimento, os valores mínimos diminuíram conforme o número de requisições aumentava, o tempo máximo aumentou conforme as requisições aumentavam, a média se manteve constante, e o desvio padrão diminuiu. A maior distinção entre os dois experimentos está no tempo máximo para uma requisição, que diminuiu de 43,3394ms para 18,1358ms.

Tabela 7 – MQTT - Taxa de requisições aceitas

Cenário	Sequencial	Exponencial
1	54%	60%
2	67%	61%
3	62%	35%
4	56%	42%
5	56%	47%

A Tabela 7 apresenta a porcentagem das requisições que chegaram no ESP32 e que foram publicadas no tópico *"lasdpc/status/check"* e lidas pelo K6. Observamos que os valores em porcentagem para os valores sequenciais variam de 54% a 67%. Enquanto que para os valores exponenciais existe uma variação entre 35% e 61%. A maior taxa de leitura é apresentada no Cenário 3 do 1º Experimento e a menor taxa de leitura é apresentada no Cenário 3 do 2º Experimento. Novos testes são necessários para determinar os motivos desses resultados.

A Figura 20 apresenta as distribuições do tempo de resposta HTTP do 1º experimento. Os dados com as médias dos 100 resultados (10 resultados por VM) foram utilizados para criar esses gráficos. Observamos que o limite superior e inferior para o Cenário 1 atingem o mínimo e o máximo em comparação com os outros cenários, que a mediana está em torno de 3,1 e o 1º Quartil fica em torno de 3,0 e o 3º Quartil em torno de 3,35. Notamos que a cada cenário, os limites superiores, inferiores, Q1 e Q3 diminuem. Ou seja, as médias estão mais próximas do valor da mediana.

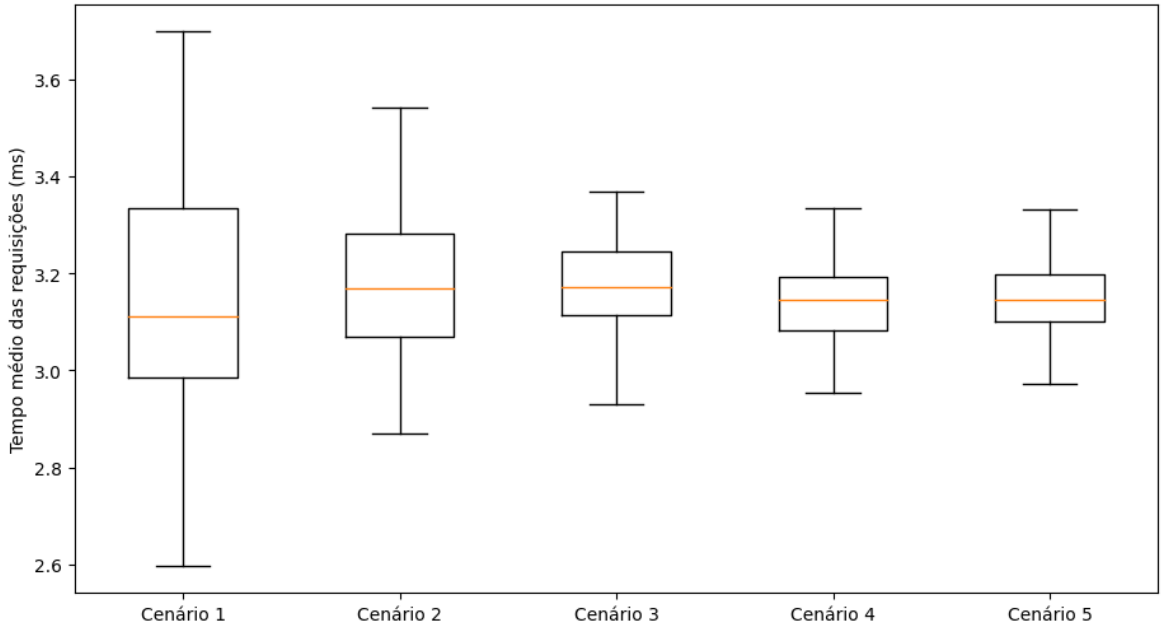

Figura 20 – Distribuição do tempo de resposta HTTP para o 1º Experimento

Para o 2º Experimento, apresentamos os gráficos na Figura 21. Observamos que o tempo médio de resposta das requisições HTTP diminui à medida que a quantidade de requisições

aumenta. É interessante observar que o maior limite superior e o menor limite inferior têm uma diferença de 0.7ms, ou seja, as requisições HTTP nestes cenários têm baixa variação de tempo de resposta.

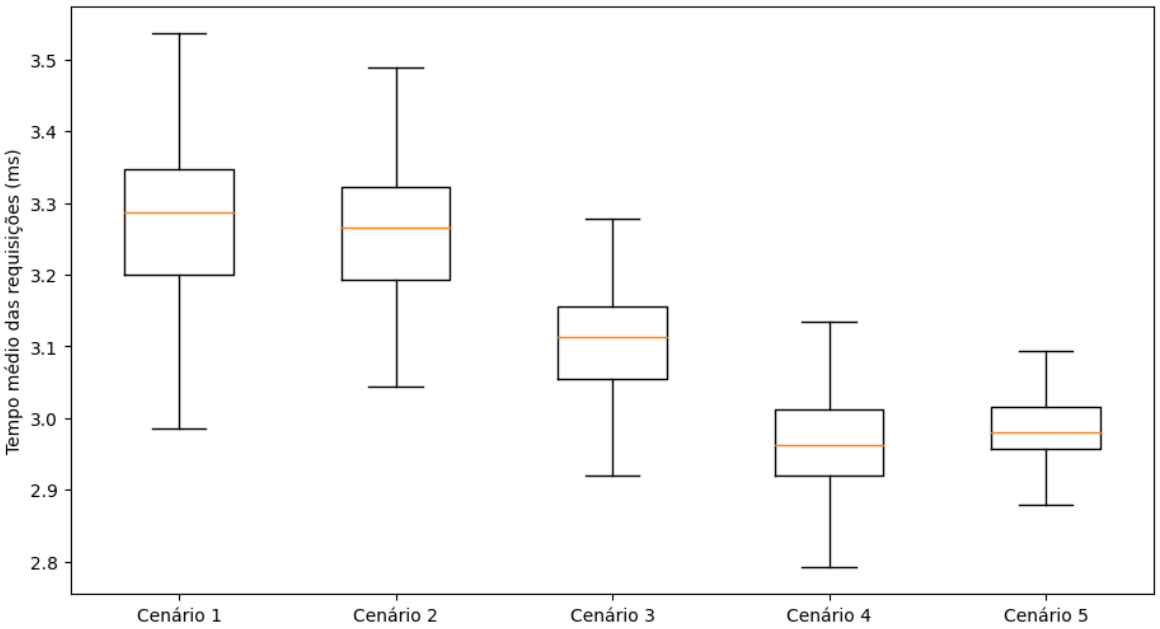

Figura 21 – Distribuição do tempo de resposta HTTP para o 2º Experimento

A Figura 22 apresenta o 1º e o 2º Experimento lado a lado. Percebemos que para o Cenário 1 o 1º Experimento têm uma limite inferior maior que o limite inferior do 2º Experimento, o mesmo ocorre para o limite superior. Observa-se também que o Q1 e Q3 do 1º Experimento do Cenário 1 têm uma variação de aproximadamente 0,4ms enquanto o 2º Experimento têm uma diferença de variaão de 0,2ms, ou seja, o tempo de resposta para o 2º Experimento têm médias mais próximas umas das outras. Porém a média do tempo de resposta do 2º Experimento é mais lenta que a média de tempo de reposta para o 1º Experimento.

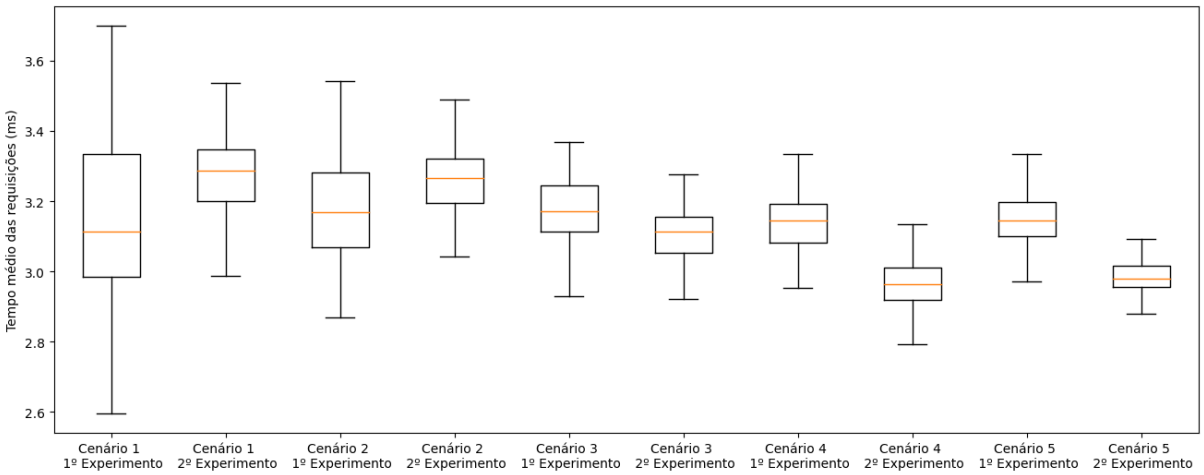

Figura 22 – Comparação do tempo de resposta entre do 1º Experimento com o 2º Experimento

Já para o Cenário 2 temos um exemplo de tempo de resposta médio das requisições semelhantes ao do Cenário 1. Vemos que a média do tempo de reposta do 1º Experimento é menor que a média do 2º Experimento. O 1º Experimento do Cenário 2 também apresenta os limites superior e inferior maiores que os apresentados pelo 2º Experimento. E as variações de médias para o 1º Experimento são maiores que as variações para o 2º Experimento.

No Cenário 3 podemos concluir que a média de tempo de resposta entre os dois experimento são parcialmente iguais. Existindo um diferença de limite inferior menor que 0,1ms, uma diferença de limite superior entre 0,1ms e os Q1 e Q3 dos dois experimento se igualando em alguns pontos.

Os Cenários 4 e 5 apresentam o 2º Experimento com um tempo de resposta das requisições inferiores ao 1º Experimento. Essa situação é inversa aos Cenários 1 e 2. Observa-se também que para os dois cenários a variação do tempo médio é inferior ao apresentado nos primeiro 2 cenários. Como a quantidade de requisições cresce o tempo médio de resposta deveria acompanhar o aumento do número de requisições. Para uma resposta mais conclusiva, novos experimentos devem ser realizados para uma conclusão mais precisa.

6.5 Considerações Finais

Nesta seção, apresentamos análises sobre o tempo de resposta de requisições HTTP e a análise da porcentagem de mensagens publicadas e lidas para o modelo arquitetural proposto. Observamos que a média das requisições HTTP ficam em torno de 3ms e 3,5ms. Além disso, foi possível observar que, conforme a carga aumenta, o tempo de requisição mínima diminui, o tempo de requisição máxima aumenta, a média se mantém constante e o desvio padrão diminui.

CAPÍTULO

7

CONCLUSÃO

Trouxemos uma discussão sobre como o tamanho da arquitetura em IoT influencia na organização dos componentes e na organização arquitetural. Além disso, abordamos uma visão dos caminhos que os dados trafegam e alguns dos possíveis eventos que podem ocorrer em aplicações reais.

Propusemos e apresentamos arquiteturas para aplicações de IoT de pequeno, médio e grande porte, ao mesmo tempo em que discutimos os componentes que compõem cada uma dessas arquiteturas. Mostramos que a complexidade de hardware e software aumenta à medida que a aplicação cresce em tamanho, resultando em aplicações de pequena escala mais simples e aplicações de grande escala mais complexas.

Demonstramos fluxogramas que exibem como os dados se comportam dentro dessas aplicações. Observa-se que os dados surgem a partir da leitura de um sensor que os transporta para unidades de processamento mais especializadas, como microcontroladores e servidores. Essas unidades de processamento mais especializadas processam os dados e os armazenam em um banco de dados. Em aplicações mais complexas, os dados gerados são analisados para gerar ações no ambiente por meio de atuadores.

Um protótipo para arquitetura de médio porte foi proposto. A partir dos resultados obtidos, pudemos observar que ele é capaz de lidar com requisições constantes e exponenciais. O Apache Kafka e o MQTT tornam o sistema mais dinâmico, orientando-o a eventos. Podemos classificar a leitura de dados e a execução de ações no ambiente como eventos interpretados pelo modelo arquitetural. Esses eventos geram ações e reações na aplicação.

O presente trabalho foi realizado com o apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

7.1 Trabalhos Publicados

"Computational Infrastructure Design to Support Applications in Intelligent Building Environments – Case Study Focused on Intelligent Laboratories" (RIBEIRO *et al.*, 2021) com o ISBN 978-1-907240-71-3, foi apresentado na conferência: CIB International Conference On Smart Built Environment (CIB ISCBE) em dezembro de 2021. Nele apresentamos os conceitos de arquiteturas de pequeno, médio e grande porte e modelos para as arquiteturas de pequeno e médio porte. Este artigo serviu como ponto de partida para o desenvolvimento deste trabalho.

Trabalhamos em conjunto com a aluna de doutorado, agora doutora, Beatriz Campos Fialho do Instituto de Arquitetura e Urbanismo (IAU) para a publicação do trabalho "Development of a BIM and IoT-Based Smart Lighting Maintenance System Prototype for Universities' FM Sector" (FIALHO *et al.*, 2022). Neste trabalho realizamos a integração de um modelo Modelagem da Informação da Construção (BIM) com uma arquitetura de IoT para a leitura de dados do ambiente para manutenção preventiva. Este trabalho foi publicado na Revista *Buildings* em 2022 com o ISSN 2075-5309.

Com a conclusão desta dissertação, estamos preparando outro artigo para publicação no MDPI, focalizando novos experimentos para o modelo proposto. O nome deste trabalho será "Synthesis of data flows in small, medium, and large-scale environments in the context of the Internet of Things" com previsão de finalização para dezembro de 2023.

CAPÍTULO

8

TRABALHOS FUTUROS

Este trabalho serve como ponto de partida para o desenvolvimento de uma série de estudos futuros. É importante ressaltar que as arquiteturas genéricas podem ser flexibilizadas mediante a adição ou remoção de componentes, de acordo com as tendências emergentes e as descobertas de novos estudos bibliográficos. Dado o dinamismo inerente à área de IoT, torna-se imperativo realizar adaptações contínuas no modelo arquitetural conforme surgem novas tecnologias e desafios.

O modelo proposto pode ser estendido para uma arquitetura de grande escala com a adição de ferramentas para processamento de dados em tempo real, ferramentas de monitoramento, adição de algoritmos de IA para o controle de ambiente e detecção da qualidade dos dados apresentados, bem como uma expansão dos protocolos para atuadores e sensores.

Diferentes experimentos podem ser realizados no modelo arquitetural proposto. Estes experimentos podem variar de mudanças nos dados de entrada à análise de processos intermediários. Essas análises de processos intermediários iriam verificar qual o desempenho dos diferentes protocolos, ou seja, a análise do tempo de entrega de mensagens entre a publicação do HTTP para o Apache Kafka, do Apache Kafka para o MQTT e do MQTT para o Apache Kafka.

O modelo arquitetural pode ser ainda empregado como gerenciador de aplicações para permitir pesquisas de desempenho, segurança e computação de borda. Isso pode ser feito com o aproveitamento do que foi desenvolvido para a adição de novos componentes e testes nesses componentes. Protocolos de segurança podem ser testados em dispositivos físicos integrados e novos dispositivos de borda computacional podem ser adicionados para testes arquiteturais nesses dispositivos, por exemplo.

Novos estudos de inteligência artificial para IoT podem ser realizados a partir dos dados armazenados pelos sensores. Esses novos estudos podem explorar um agrupamento dos dados dos sensores para gerar informações relevantes sobre o ambiente que podem fornecer economia de energia ou preferências pessoais dos usuários de um determinado ambiente.

Diferentes caminhos podem ser explorados a partir desta dissertação. Alguns caminhos são apresentados acima e outros podem ser propostos.

REFERÊNCIAS

AMADEO, M.; MOLINARO, A.; PARATORE, S. Y.; ALTOMARE, A.; GIORDANO, A.; MASTROIANNI, C. A cloud of things framework for smart home services based on information centric networking. In: **2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)**. [S.l.: s.n.], 2017. p. 245–250. Citado nas páginas 13, 41, 81 e 85.

BANSAL, S.; KUMAR, D. Iot ecosystem: A survey on devices, gateways, operating systems, middleware and communication. **International Journal of Wireless Information Networks**, 2020. Cited By 1. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85079720826&doi=10.1007%2fs10776-020-00483-7&partnerID=40&md5=e61b0e652e0bc7d6123714ba402958f5>. Citado nas páginas 23, 25 e 31.

BHARADWAJ, A. S.; REGO, R.; CHOWDHURY, A. Iot based solid waste management system: A conceptual approach with an architectural solution as a smart city application. In: **2016 IEEE Annual India Conference (INDICON)**. [S.l.: s.n.], 2016. p. 1–6. Citado nas páginas 13, 42, 82 e 89.

BIYIK, C. Smart Cities in Turkey: Approaches, Advances and Applications with Greater Consideration for Future Urban Transport Development. **Energies**, v. 12, 2019. ISSN 1996-1073. Citado na página 32.

CAI, Q.; WANG, H.; LI, Z.; LIU, X. A survey on multimodal data-driven smart healthcare systems: Approaches and applications. **IEEE Access**, IEEE Access, v. 7, p. 133583–133599, 2019. ISSN 2169-3536. Citado na página 32.

CATARINUCCI, L.; DONNO, D. de; MAINETTI, L.; PALANO, L.; PATRONO, L.; STEFANIZZI, M. L.; TARRICONE, L. An iot-aware architecture for smart healthcare systems. **IEEE Internet of Things Journal**, v. 2, n. 6, p. 515–526, 2015. Citado nas páginas 13, 41, 82 e 88.

DAISSAOUI, A.; BOULMAKOUL, A.; KARIM, L.; LBATH, A. Iot and big data analytics for smart buildings: A survey. **Procedia Computer Science**, Procedia Computer Science, v. 170, p. 161–168, 2020. ISSN 1877-0509. Citado na página 32.

FIALHO, B. C.; CODINHOTO, R.; FABRICIO, M. M.; ESTRELLA, J. C.; RIBEIRO, C. M. N.; BUENO, J. M. d. S.; TORREZAN, J. P. D. Development of a bim and iot-based smart lighting maintenance system prototype for universities' fm sector. **Buildings**, v. 12, n. 2, 2022. ISSN 2075-5309. Disponível em: <https://www.mdpi.com/2075-5309/12/2/99>. Citado na página 74.

Fortune Business Insights. resreport, **Internet of Things (IoT) Market Size, Share Covid-19 Impact Analysis, By Component (Platform, Solution and Service), By Platform (Device Management, Cloud Platform and Network Management), By Solution (Real-Time Streaming Analytics, Security, Data Management, Remote Monitoring), By End-Use (BFSI, Rerail, Government, Healthcare, Manufacturing, Agriculture, Sustainable Energy) and Regional Forecast, 2020 - 2027**. 2020. Disponível em: <https://www.fortunebusinessinsights.com/industry-reports/internet-of-things-iot-market-100307>. Acesso em: 22 fev. 2021 às 13:45. Citado na página 26.

GAIKWAD, P. P.; GABHANE, J. P.; GOLAIT, S. S. A survey based on smart homes system using internet-of-things. In: **2015 International Conference on Computation of Power, Energy, Information and Communication (ICCPEIC)**. [S.l.: s.n.], 2015. p. 0330–0335. Citado na página 31.

GUPTA, B.; QUAMARA, M. An overview of internet of things (iot): Architectural aspects, challenges, and protocols. **Concurrency and Computation: Practice and Experience**, v. 32, n. 21, p. e4946, 2020. E4946 CPE-18-0159.R1. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.4946>. Citado na página 26.

IQBAL, A.; ULLAH, F.; ANWAR, H.; KWAK, K. S.; IMRAN, M.; JAMAL, W.; RAHMAN, A. ur. Interoperable internet-of-things platform for smart home system using web-of-objects and cloud. **Sustainable Cities and Society**, v. 38, p. 636–646, 2018. ISSN 2210-6707. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2210670717313379>. Citado nas páginas 13, 40, 41, 81 e 84.

IQBAL, J.; KHAN, M.; TALHA, M.; FARMAN, H.; JAN, B.; MUHAMMAD, A.; KHATTAK, H. A. A generic internet of things architecture for controlling electrical energy consumption in smart homes. **Sustainable Cities and Society**, v. 43, p. 443–450, 2018. Citado nas páginas 13, 40, 41, 81 e 84.

K6. 2023. <https://k6.io/docs/>. Acessado: 20/10/2023. Citado na página 65.

KHOI, N. M.; SAGUNA, S.; MITRA, K.; HLUND, C. Irehmo: An efficient iot-based remote health monitoring system for smart regions. In: **2015 17th International Conference on E-health Networking, Application Services (HealthCom)**. [S.l.: s.n.], 2015. p. 563–568. Citado nas páginas 14, 42, 82 e 89.

KIRIMTAT, A.; KREJCAR, O.; KERTESZ, A.; TASGETIREN, M. F. Future trends and current state of smart city concepts: A survey. **IEEE Access**, v. 8, p. 86448–86467, 2020. Citado na página 33.

KUROSE, J.; ROSS, K.; MARQUES, A.; ZUCCHI, W. **Redes de computadores e a Internet: uma abordagem top-down**. [S.l.]: Pearson Addison Wesley, 2013. ISBN 978-85-430-1443-2. Citado nas páginas 26 e 28.

LOHACHAB, A. Bootstrapping urban planning: Addressing big data issues in smart cities. In: ____. [S.l.: s.n.], 2019. p. 217–246. ISBN 9781522597445. Citado na página 32.

LUETH, K. L. **IoT 2019 in Review: The 10 Most Relevant IoT Developments of the Year**. [S.l.], 2020. Disponível em: <https://iot-analytics.com/iot-2019-in-review>. Citado na página 26.

MALCHE, T.; MAHESHWARY, P. Internet of things (iot) for building smart home system. In: **2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)**. [S.l.: s.n.], 2017. p. 65–70. Citado nas páginas 13, 39, 40, 46, 81 e 83.

MARRA, A. L.; MARTINELLI, F.; MORI, P.; SARACINO, A. Implementing usage control in internet of things: A smart home use case. In: **2017 IEEE Trustcom/BigDataSE/ICESS**. [S.l.: s.n.], 2017. p. 1056–1063. Citado na página 81.

MONTORI, F.; BEDOGNI, L.; BONONI, L. A collaborative internet of things architecture for smart cities and environmental monitoring. **IEEE Internet of Things Journal**, v. 5, n. 2, p. 592–605, 2018. Citado nas páginas 14, 43, 82 e 92.

Ngu, A. H.; Gutierrez, M.; Metsis, V.; Nepal, S.; Sheng, Q. Z. Iot middleware: A survey on issues and enabling technologies. **IEEE Internet of Things Journal**, v. 4, n. 1, p. 1–20, 2017. Citado na página 26.

OASIS. **MQTT Version 5.0**. [S.l.: s.n.], 2019. Citado na página 28.

OUADDAH, A.; KALAM, A. A. E.; OUAHMAN, A. A. Fairaccess: a new blockchain-based access control framework for the internet of things. **Secur. Commun. Networks**, v. 9, p. 5943–5964, 2016. Citado nas páginas 42 e 82.

PARK, E.; KIM, S.; KIM, Y.; KWON, S. J. Smart home services as the next mainstream of the ict industry: determinants of the adoption of smart home services. **Universal Access in the Information Society**, Universal Access in the Information Society, v. 17, n. 1, p. 175–190, 2018. ISSN 1615-5289. Citado na página 31.

PARK, J. ho; SALIM, M. M.; JO, J.; SICATO, J. C. S.; RATHORE, S.; PARK, J. H. Ciot-net: a scalable cognitive iot based smart city network architecture. **Human-centric Computing and Information Sciences**, v. 9, p. 1–20, 2019. Citado nas páginas 14, 44, 82 e 93.

PETNIK, J.; VANUS, J. Design of smart home implementation within iot with natural language interface. **IFAC-PapersOnLine**, v. 51, n. 6, p. 174–179, 2018. ISSN 2405-8963. 15th IFAC Conference on Programmable Devices and Embedded Systems PDeS 2018. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2405896318308954>. Citado nas páginas 13, 41, 81 e 85.

PULIAFITO, A.; TRICOMI, G.; ZAFEIROPOULOS, A.; PAPAVASSILIOU, S. Smart cities of the future as cyber physical systems: Challenges and enabling technologies. **Sensors**, v. 21, n. 10, 2021. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/21/10/3349>. Citado na página 33.

RAHMANI, A. M.; GIA, T. N.; NEGASH, B.; ANZANPOUR, A.; AZIMI, I.; JIANG, M.; LILJEBERG, P. Exploiting smart e-health gateways at the edge of healthcare internet-of-things. **Future Gener. Comput. Syst.**, Elsevier Science Publishers B. V., NLD, v. 78, n. P2, p. 641–658, jan. 2018. ISSN 0167-739X. Disponível em: <https://doi.org/10.1016/j.future.2017.02.014>. Citado nas páginas 13, 41, 42, 49, 82 e 87.

RATHORE, M. M.; AHMAD, A.; PAUL, A.; RHO, S. Urban planning and building smart cities based on the internet of things using big data analytics. **Computer Networks**, v. 101, p. 63–80, 2016. ISSN 1389-1286. Industrial Technologies and Applications for the Internet of Things. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1389128616000086>. Citado nas páginas 13, 14, 43, 82 e 91.

RATHORE, M. M.; PAUL, A.; HONG, W.-H.; SEO, H.; AWAN, I.; SAEED, S. Exploiting iot and big data analytics: Defining smart digital city using real-time urban data. **Sustainable Cities and Society**, v. 40, p. 600–610, 2018. ISSN 2210-6707. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2210670717309782>. Citado nas páginas 43 e 82.

RIBEIRO, C. M. N.; ESTRELLA, J. C.; BUENO, J. M. d. S.; TRAZZI, B. M.; TORREZAN, J. P. D.; FIALHO, B. C. **Development of a BIM and IoT-Based Smart Lighting Maintenance System Prototype for Universities' FM Sector**. [s.n.], 2021. 136-145 p. ISBN 978-1-907240-71-3. Disponível em: <https://www.leedsbeckett.ac.uk/-/media/files/events/cib-conference/cib-conference-tg96-2021-proceedings.pdf>. Citado na página 74.

SUNHARE, P.; CHOWDHARY, R. R.; CHATTOPADHYAY, M. K. Internet of things and data mining: An application oriented survey. **Journal of King Saud University - Computer and Information Sciences**, Journal of King Saud University - Computer and Information Sciences, 2020. ISSN 1319-1578. Citado nas páginas 13 e 30.

THADUANGTA, B.; CHOOMJIT, P.; MONGKOLVESWITH, S.; SUPASITTHIMETHEE, U.; FUNILKUL, S.; TRIYASON, T. Smart healthcare: Basic health check-up and monitoring system for elderly. In: ____. [S.l.: s.n.], 2016. Citado na página 32.

The Internet Society. **User Datagram Protocol**. 1980. Disponível em: <https://datatracker.ietf.org/doc/html/rfc768>. Acesso em: 19 ago. 2021 às 10:12. Citado na página 29.

____. **TRANSMISSION CONTROL PROTOCOL**. 1981. Disponível em: <https://datatracker.ietf.org/doc/html/rfc793>. Acesso em: 19 ago. 2021 às 09:53. Citado na página 29.

____. **Hypertext Transfer Protocol – HTTP/1.1**. 1999. Disponível em: <https://tools.ietf.org/html/rfc2616>. Acesso em: 26 fev. 2021 às 09:51. Citado na página 28.

VISWANATH, S. K.; YUEN, C.; TUSHAR, W.; LI, W.-T.; WEN, C.-K.; HU, K.; CHEN, C.; LIU, X. System design of the internet of things for residential smart grid. **IEEE Wireless Communications**, v. 23, n. 5, p. 90–98, 2016. Citado nas páginas 14, 44, 50, 82 e 92.

WEN, Z.; HU, S.; De Clercq, D.; BECK, M. B.; ZHANG, H.; ZHANG, H.; FEI, F.; LIU, J. Design, implementation, and evaluation of an internet of things (iot) network system for restaurant food waste management. **Waste Management**, v. 73, p. 26–38, 2018. ISSN 0956-053X. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0956053X17309376>. Citado nas páginas 13, 42, 82 e 88.

ZHANG, M.; YU, T.; ZHAI, G. Smart transport system based on “the internet of things. **AMM**, 2011. Citado na página 32.

APÊNDICE

A

TABELAS COM OS TRABALHOS RELACIONADOS

Tabela 8 – Tabela de artigos de pequena escala

Autor(es)	**Título**	**Aplicação**
Malche e Maheshwary (2017)	Internet of Things (IoT) for building Smart Home System	Casas Inteligentes
Iqbal *et al.* (2018a)	Interoperable Internet-of-Things platform for smart home system using Web-of-Objects and cloud	Casas Inteligentes
Iqbal *et al.* (2018b)	A generic internet of things architecture for controlling electrical energy consumption in smart homes	Casas Inteligentes / Edifícios Inteligentes
Marra *et al.* (2017)	Implementing Usage Control in Internet of Things: A Smart Home Use Case	Casas Inteligentes
Petnik e Vanus (2018)	Design of Smart Home Implementation within IoT with Natural Language Interface	Casas Inteligentes
Amadeo *et al.* (2017)	A Cloud of Things Framework for Smart Home Services based on Information Centric Networking	Casas Inteligentes

Tabela 9 – Tabela de artigos de média escala

Autor(es)	**Título**	**Aplicação**
Rahmani *et al.* (2018)	Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things:A fog computing approach	Saúde Inteligente
Catarinucci *et al.* (2015)	An IoT-Aware Architecture for Smart Healthcare Systems	Saúde Inteligente
Ouaddah, Kalam e Ouahman (2016)	FairAccess: a new Blockchain-based access control framework for the Internet of Things	Gerenciamento do Controle de Acesso
Wen *et al.* (2018)	Design, implementation, and evaluation of an Internet of Things (IoT) network system for restaurant food waste management	Gestão de Resíduos
Bharadwaj, Rego e Chowdhury (2016)	IoT based solid waste management system: A conceptual approach with an architectural solution as a smart city application	Gestão de Resíduos
Khoi *et al.* (2015)	IReHMo: An efficient IoT-based remote health monitoring system for smart regions	Saúde Inteligente

Tabela 10 – Tabela de artigos de grande escala

Autor(es)	**Título**	**Aplicação**
Rathore *et al.* (2016)	Urban Planning and Building Smart Cities based on the Internet of Things using Big Data Analytics	Cidades Inteligentes
Rathore *et al.* (2018)	Exploiting IoT and Big Data Analytics: Defining Smart Digital City using Real-Time Urban Data	Cidades Inteligentes
Montori, Bedogni e Bononi (2018)	A Collaborative Internet of Things Architecture for Smart Cities and Environmental Monitoring	Cidades Inteligentes
Viswanath *et al.* (2016)	System design of the internet of things for residential smart grid	*Smart Grid*
Park *et al.* (2019)	CIoT-Net: a scalable cognitive IoT based smart city network architecture	Cidades Inteligentes

APÊNDICE

B

SÍNTESE DAS ARQUITETURAS DE PEQUENO PORTE

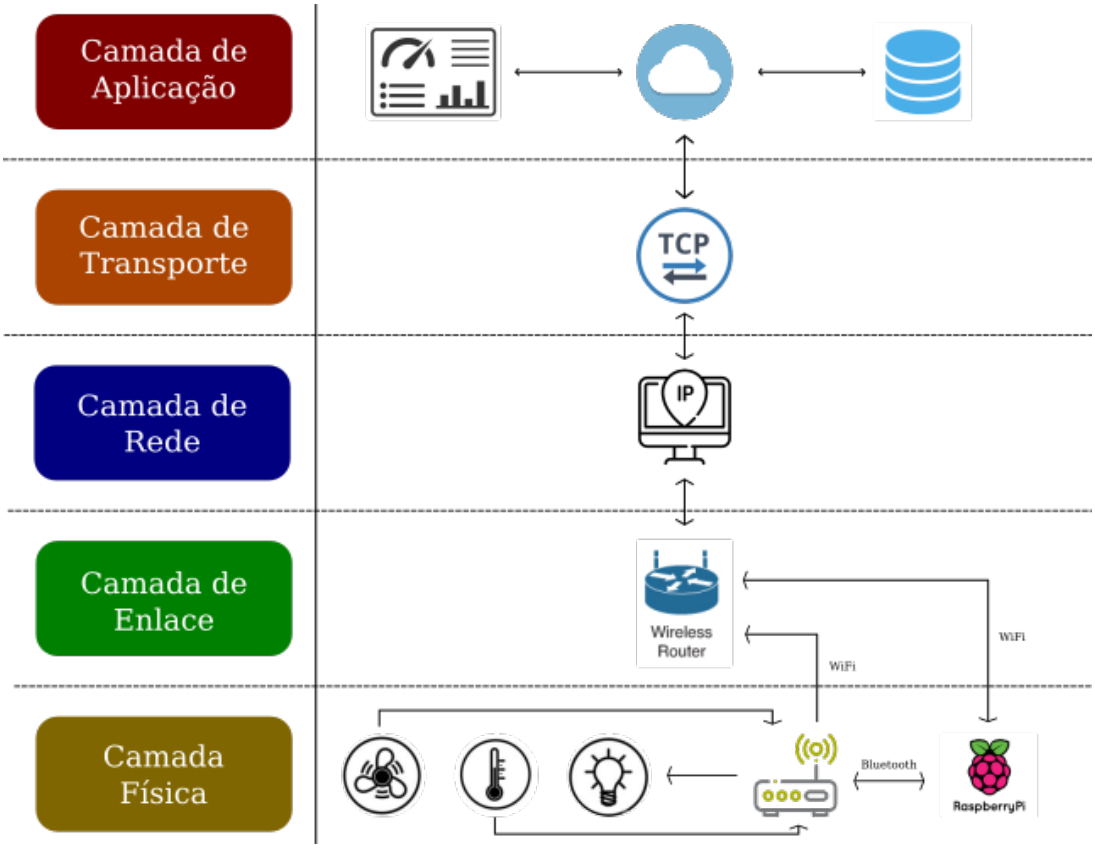

Figura 23 – Adaptado da arquitetura proposta por Malche e Maheshwary (2017)

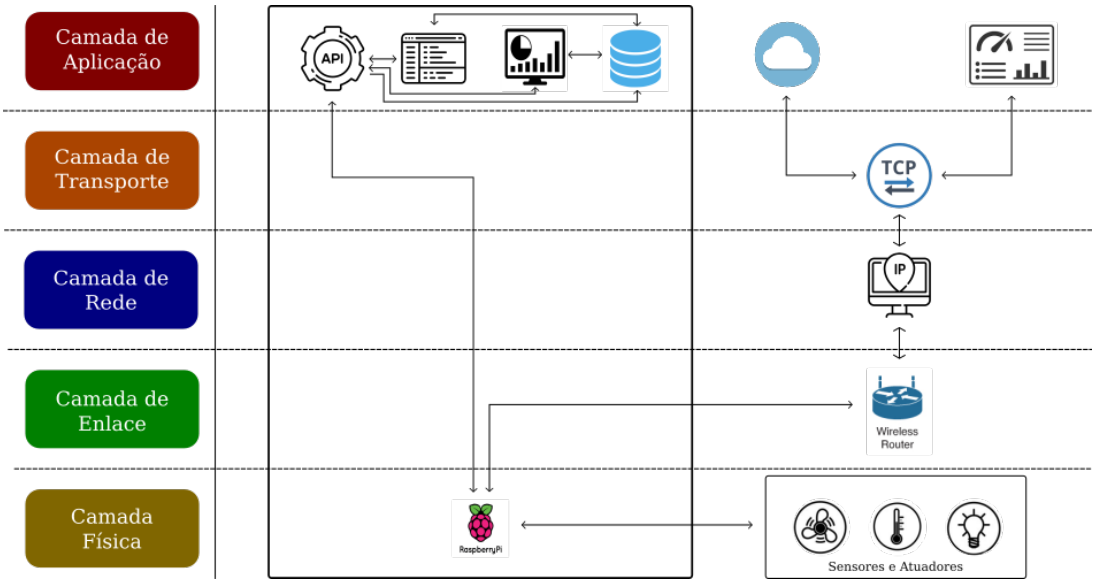

Figura 24 – Adaptado da arquitetura proposta por Iqbal *et al.* (2018a)

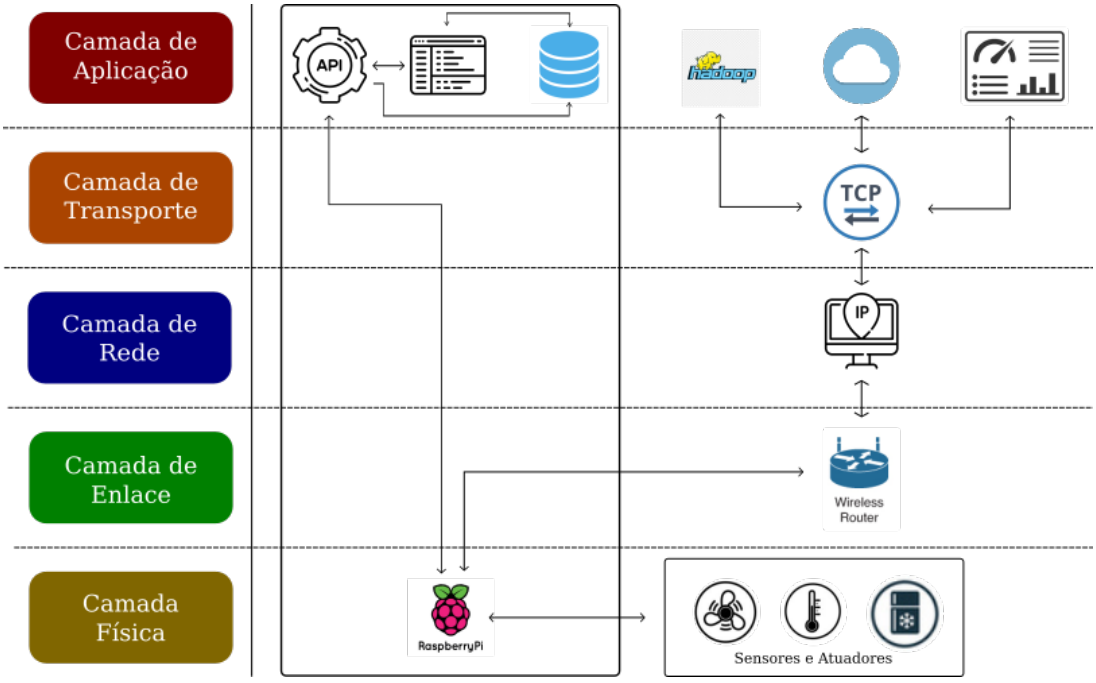

Figura 25 – Adaptado da arquitetura proposta por Iqbal *et al.* (2018b)

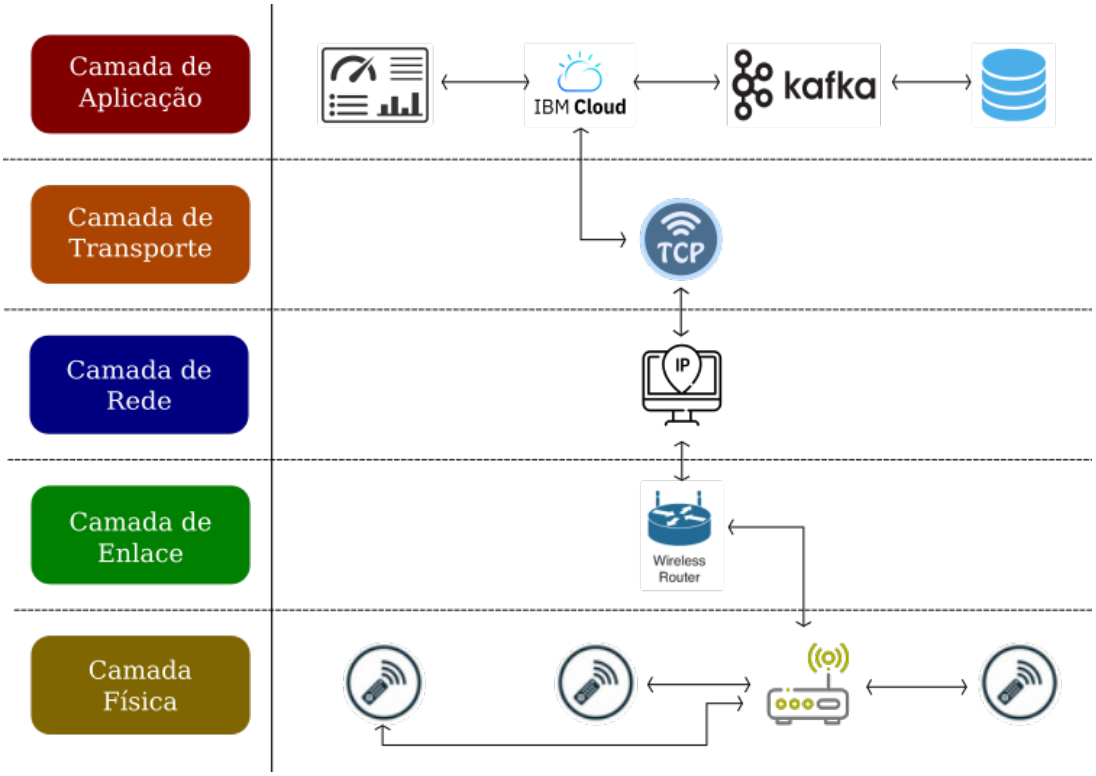

Figura 26 – Adaptado da arquitetura proposta por Petnik e Vanus (2018)

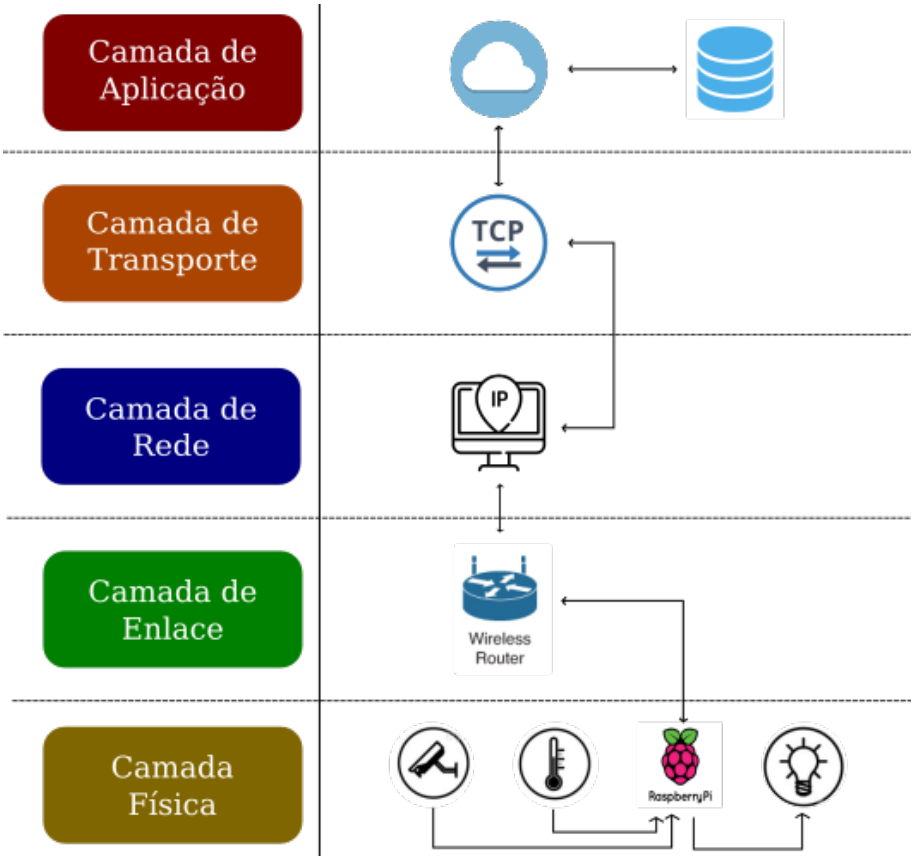

Figura 27 – Adaptado da arquitetura proposta por Amadeo *et al.* (2017)

APÊNDICE

C

SÍNTESE DAS ARQUITETURAS DE MÉDIO PORTE

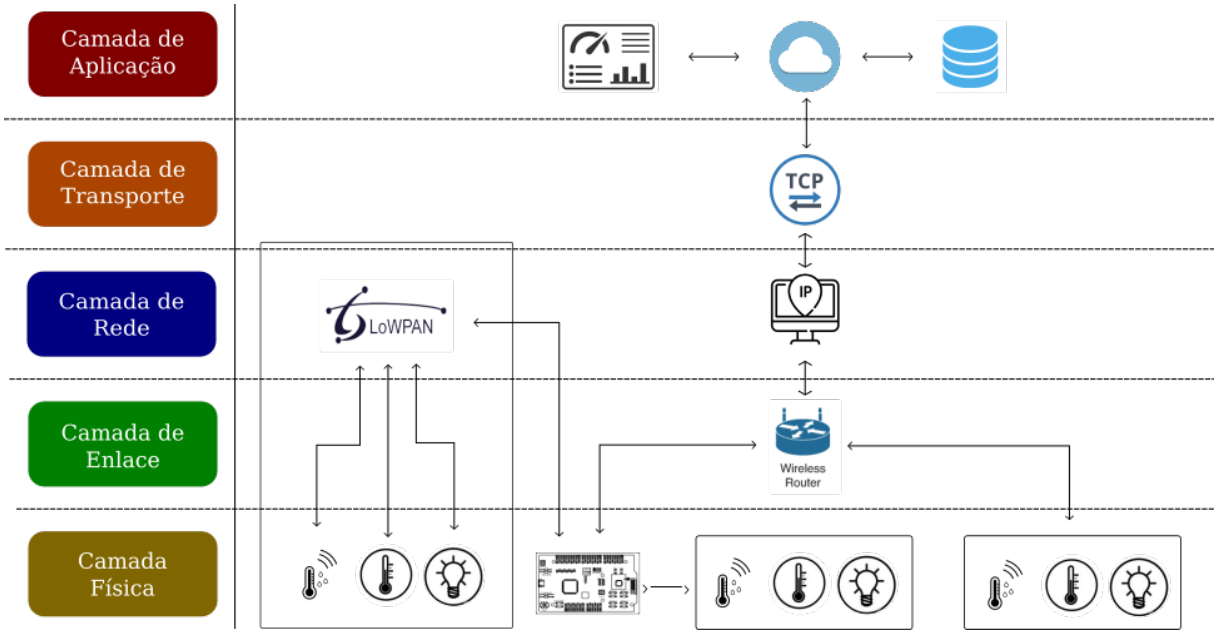

Figura 28 – Adaptado da arquitetura proposta por Rahmani *et al.* (2018)

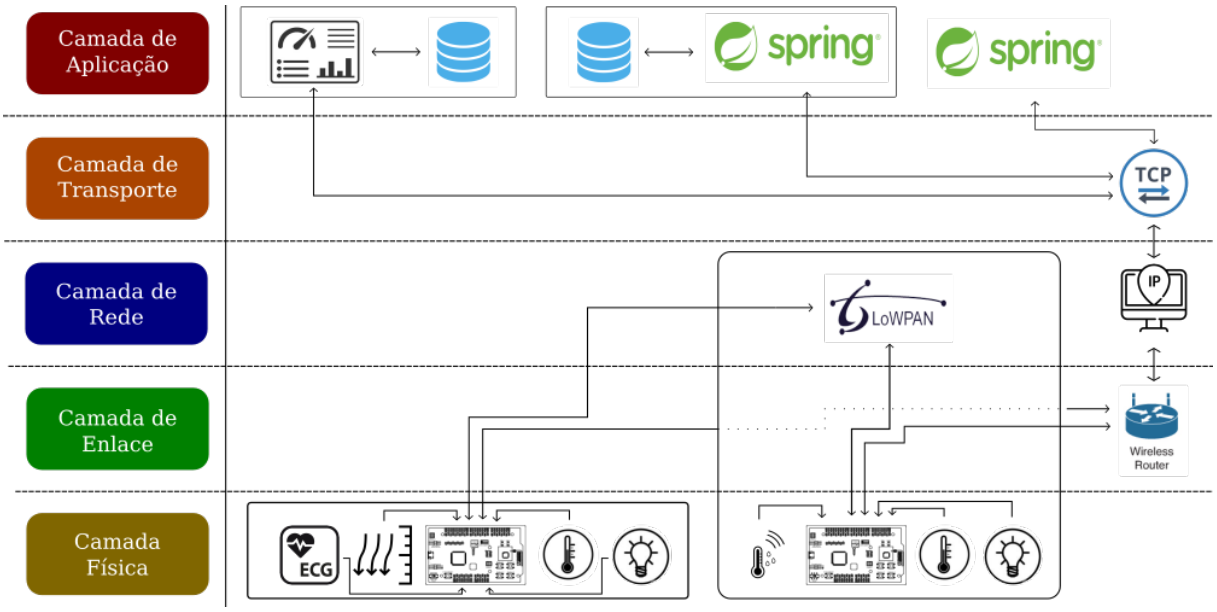

Figura 29 – Adaptado da arquitetura proposta por Catarinucci *et al.* (2015)

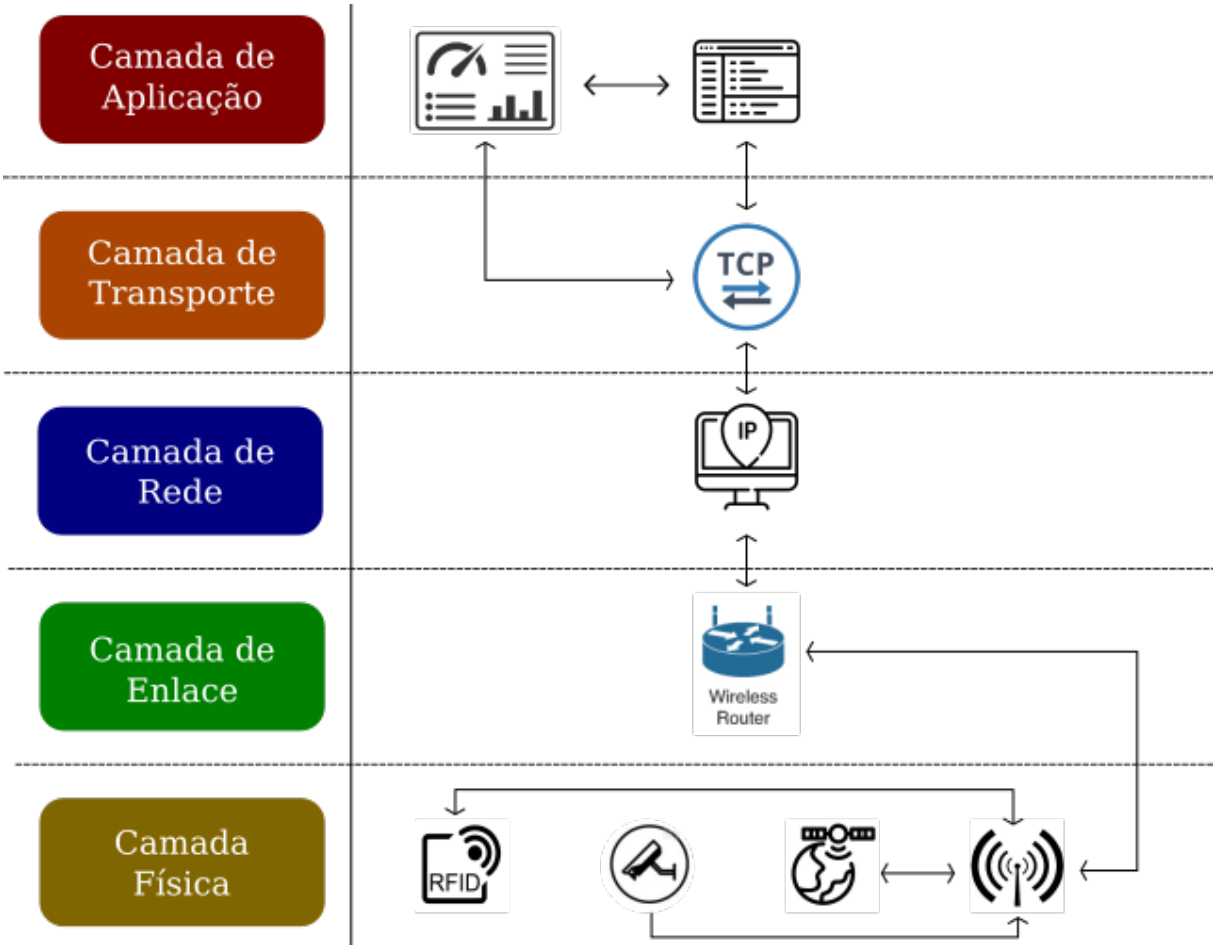

Figura 30 – Adaptado da arquitetura proposta por Wen *et al.* (2018)

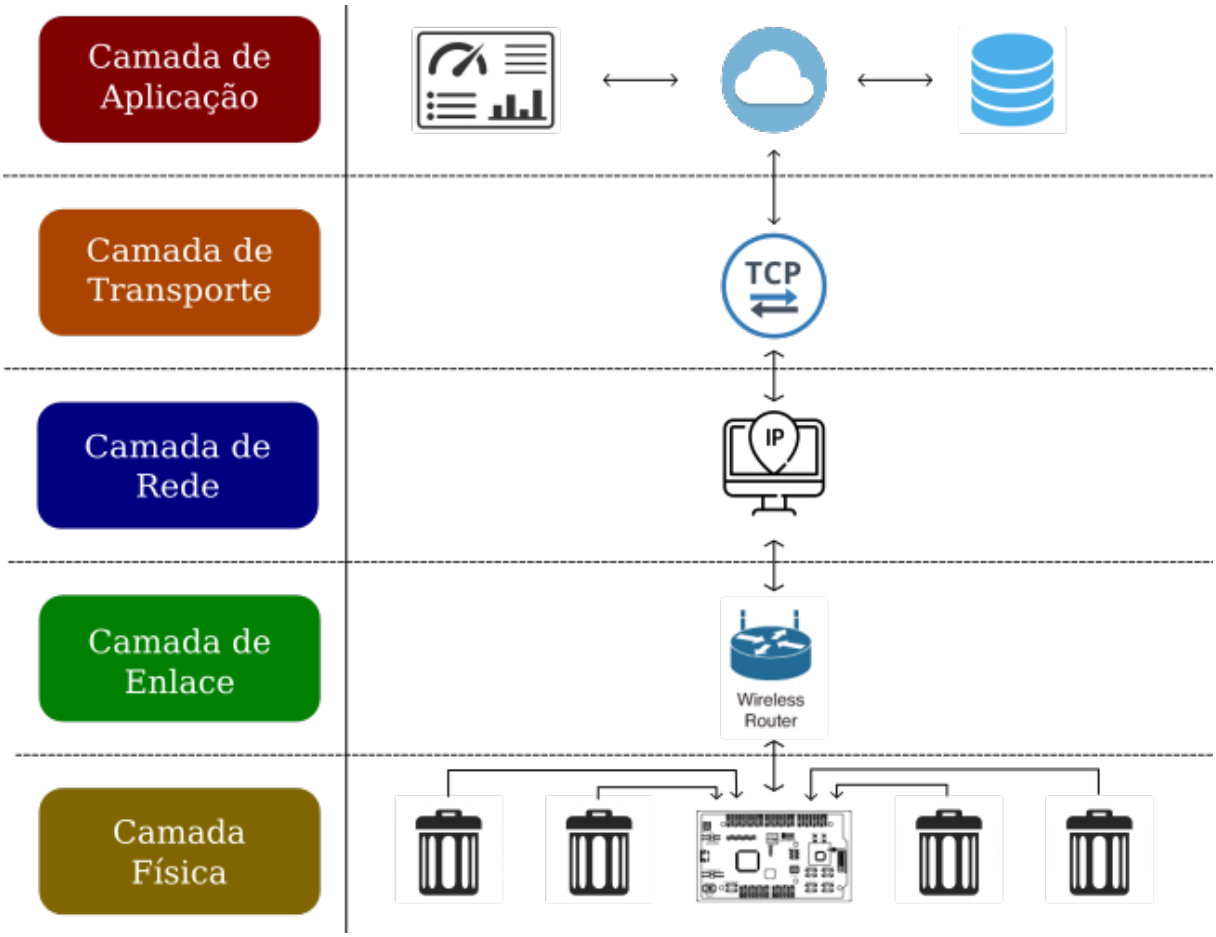

Figura 31 – Adaptado da arquitetura proposta por Bharadwaj, Rego e Chowdhury (2016)

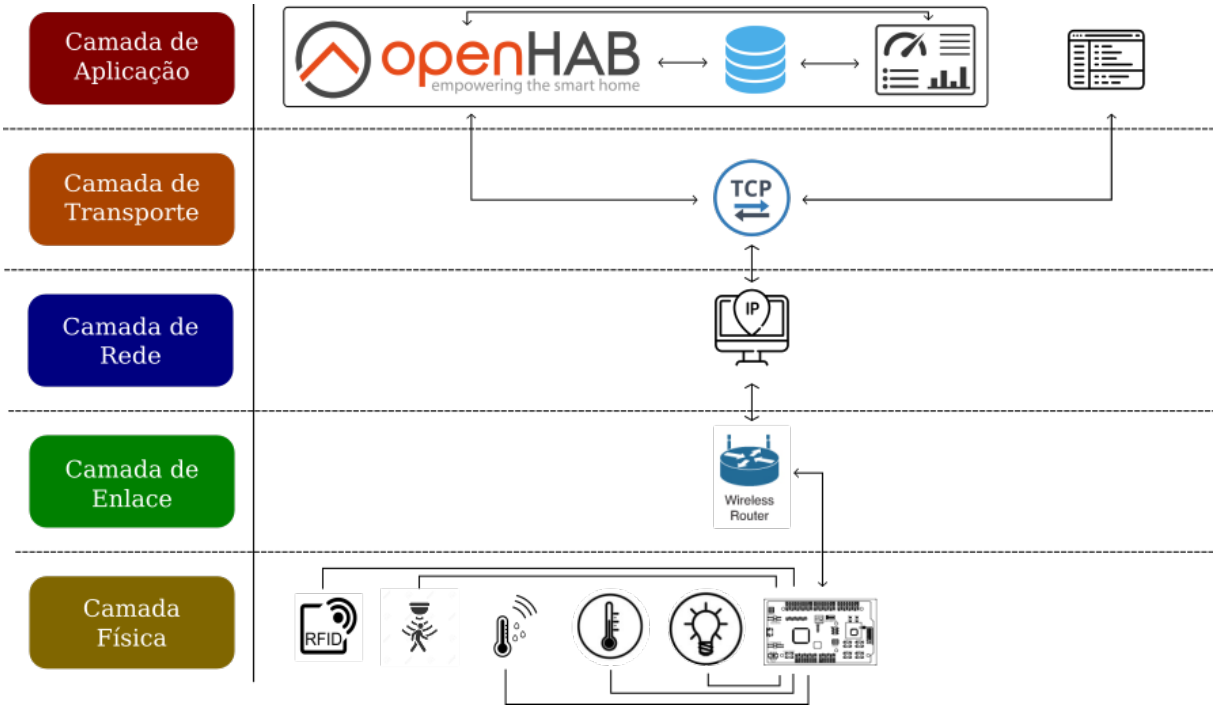

Figura 32 – Adaptado da arquitetura proposta por Khoi *et al.* (2015)

APÊNDICE

D

SÍNTESE DAS ARQUITETURAS DE GRANDE PORTE

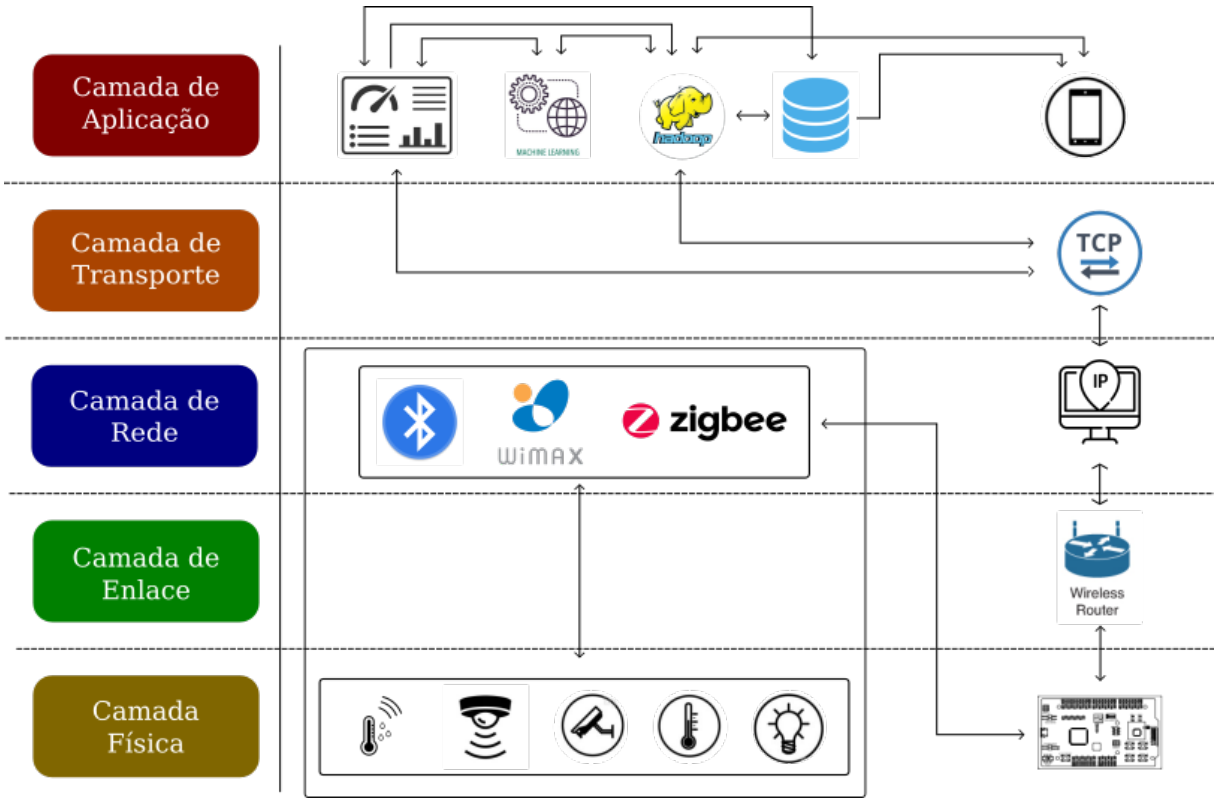

Figura 33 – Adaptado da arquitetura proposta por Rathore *et al.* (2016)

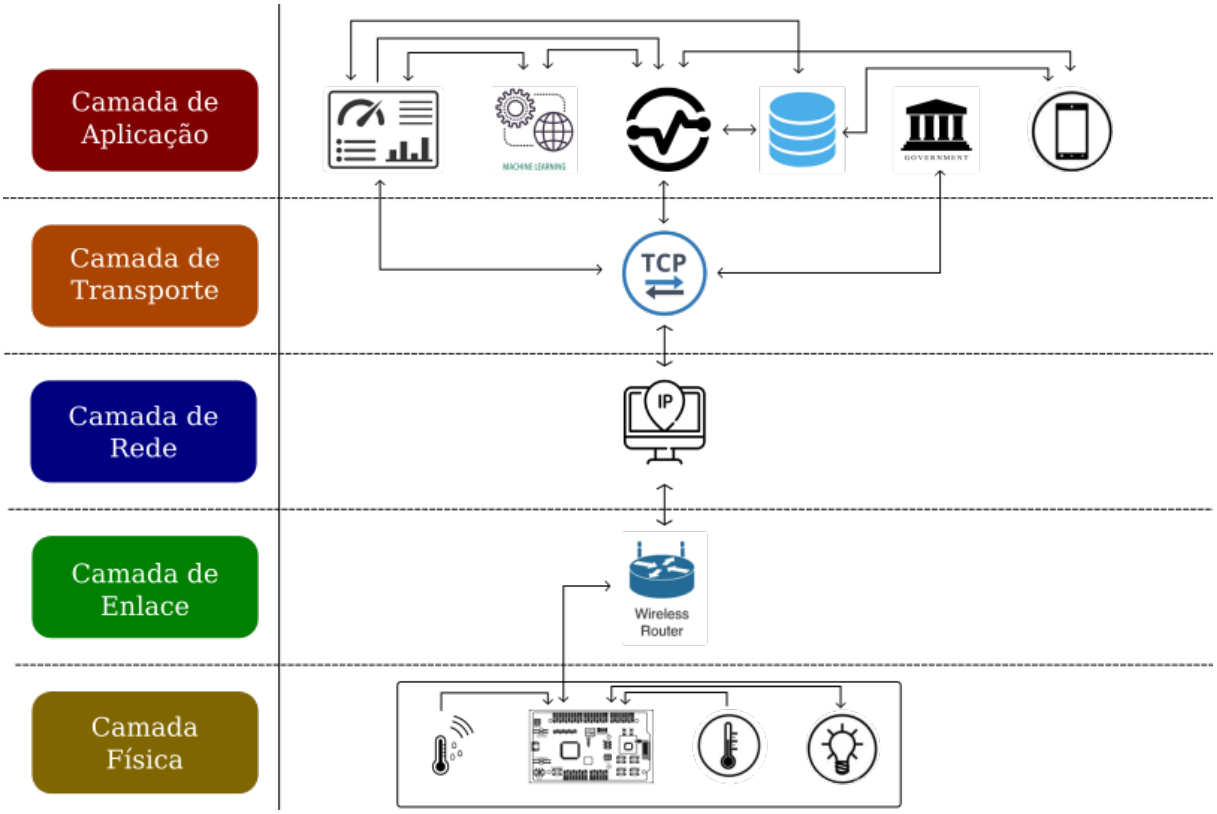

Figura 34 – Adaptado da arquitetura proposta por Montori, Bedogni e Bononi (2018)

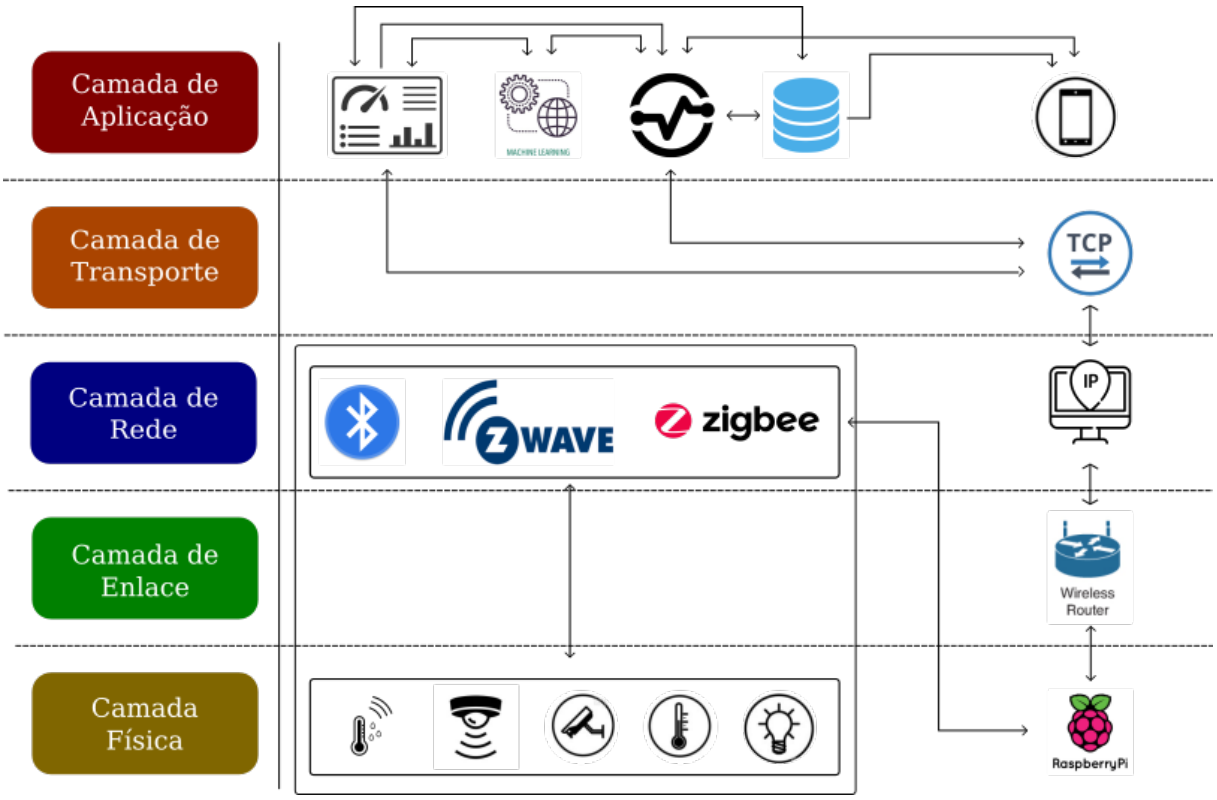

Figura 35 – Adaptado da arquitetura proposta por Viswanath *et al.* (2016)

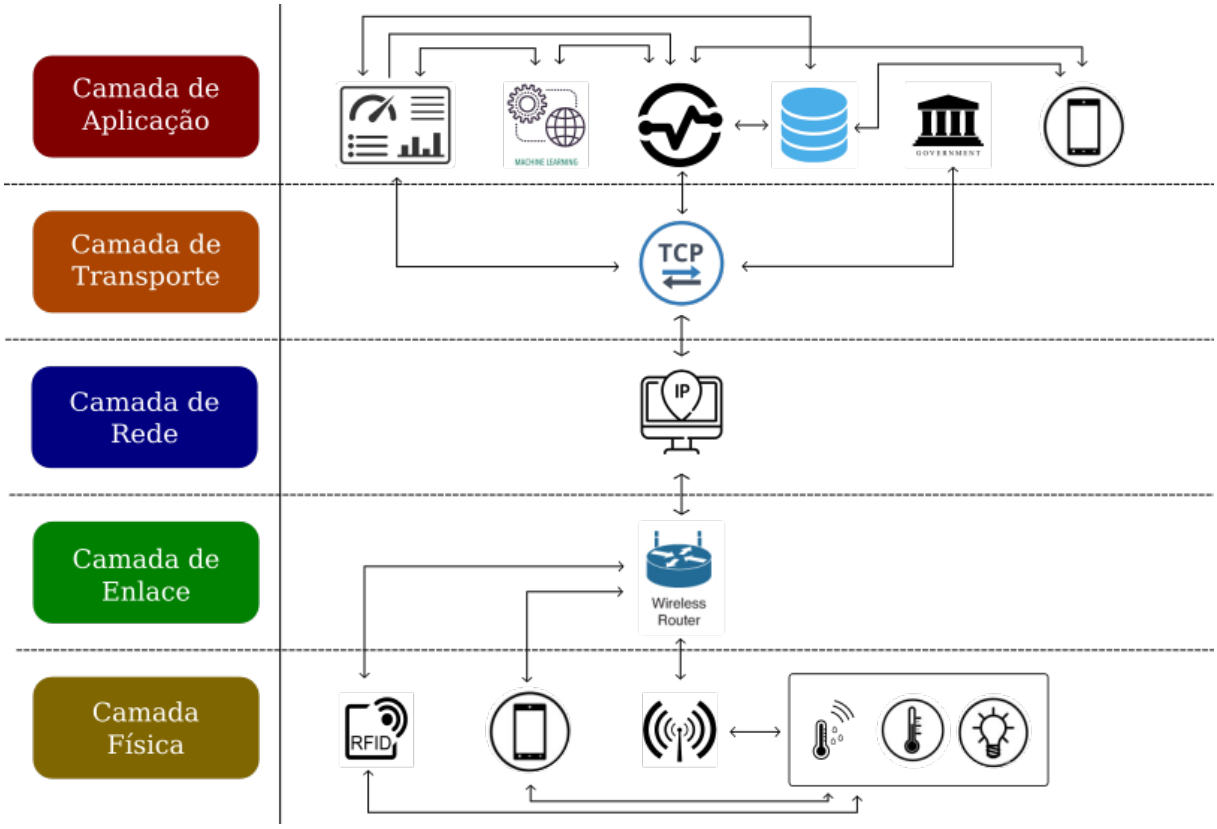

Figura 36 – Adaptado da arquitetura proposta por Park *et al.* (2019)


ICMC
USP
SÃO CARLOS