UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

FRANCISCO PAIVA KNEBEL

Designing and Implementing Digital Twins with Cloud and Edge Computing: Challenges and Opportunities

Porto Alegre
2024

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

FRANCISCO PAIVA KNEBEL

Designing and Implementing Digital Twins with Cloud and Edge Computing: Challenges and Opportunities

Thesis presented in partial fulfillment of the requirements for the degree of Master of Computer Science

Advisor: Prof. Dr. Juliano Araújo Wickboldt

Porto Alegre
2024

CIP — CATALOGING-IN-PUBLICATION

Knebel, Francisco Paiva

Designing and Implementing Digital Twins with Cloud and Edge Computing: Challenges and Opportunities / Francisco Paiva Knebel. – Porto Alegre: PPGC da UFRGS, 2024.

149 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2024. Advisor: Juliano Araújo Wickboldt.

1. Digital Twins. 2. MQTT. 3. Apache Kafka. 4. Cloud Computing. 5. Edge Computing. I. Wickboldt, Juliano Araújo. II. Título.

Dedico este trabalho às memórias de meu pai, Norberto, e minha mãe, Silvia.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Professor Juliano Araujo Wickboldt, for all the years of guidance that resulted in this work, the previous and any future ones that are surely going to be influenced by all the work done during this time.

I also acknowledge the PeTWIN project, led by Professor Mara Abel and Professor João Cesar Netto and to all other members. The project has a direct motivation link towards the developed work. PeTWIN - Digital Twins for Production Optimization and Management project[1] is a partnership between the Research Group Computing Systems for Oil E&P of INF-UFRGS and the Sirius Lab of the University of Oslo, made possible through the Financier of Studies and Projects (FINEP) and the Norwegian Research Council (RCN), co-financed by the companies Petrobras, Shell, Total Energies, CNPC and CNOOC, which make up the Libra consortium in Brazil, and the Norwegian Shell and Equinor.

[1] More information can be found at the project homepage: `https://petwin.org`

AGRADECIMENTOS

Gostaria de agradecer ao meu orientador, Professor Juliano Araujo Wickboldt, por todos os anos de orientação que resultaram neste trabalho, nos anteriores e em todos os futuros que certamente serão influenciados por todo o trabalho realizado durante esse período.

Também agradeço ao projeto PeTWIN, liderado pela Professora Mara Abel e o Professor João Cesar Netto e a todos os outros membros. O projeto tem um vínculo direto de motivação com o trabalho desenvolvido. O projeto PeTWIN - Gêmeos DIgitais para Otimização e Gerenciamento de Produção[2] é uma parceria entre o grupo Sistemas de Computação para E&P de Petróleo do INF-UFRGS e o Sirius Lab da Universidade de Oslo, viabilizada por meio da Financiadora de Estudos e Projetos (FINEP) e do Conselho de Pesquisa da Noruega (RCN), cofinanciada pelas empresas Petrobras, Shell, Total Energies, CNPC e CNOOC, que compõem o consórcio Libra no Brasil, e a Shell e Equinor norueguesas.

[2]Mais informações podem ser encontradas na página inicial do projeto: `https://petwin.org`

“If you wish to make an apple pie from scratch,
you must first invent the universe.”
— CARL SAGAN, COSMOS

ABSTRACT

Digital twins are interconnected models of a physical and a virtual system that interact in real-time. The virtual component is software that mirrors the actions of the physical counterpart, identifies issues, and autonomously resolves them without human intervention. The advent of the Internet of Things and affordable sensory devices have given rise to the contemporary digital twin that requires the ability to control their physical counterparts, necessitating a dependable connection. This research investigates the relationship between data acquisition systems and event processing in the context of cloud and edge computing digital twins. The primary challenge of this research is to devise methods to enhance the performance of data-intensive systems with escalating data input while adhering to time, cost, and complexity constraints. This work introduces a study on digital twins enabled by cloud and edge computing, experimenting with the real-time communication needs of a digital twin system deployed in a multi-component cloud and edge cluster environment. This work broadens the scope towards designing digital twin systems to operate at scale, conducting experiments using MQTT and Apache Kafka to identify potential issues for cloud and edge devices. The experiments employed a network setup using a Kubernetes cluster, unlike previous simulation approaches that solely utilized virtual machines, to achieve more authentic and precise outcomes. The experiments revealed potential issues in the scenario executions due to the brokers requiring more resources than the low-specification edge devices could offer while also deliberating on the compromises and constraints of employing Kafka for digital twins, particularly in these low-specification scenarios. Introducing an additional computation layer to a highly data-intensive application might strain the hardware available to the digital twin system. Configuring Kafka is also a complex task to manage topic partitioning, data acquisition, and replication factors. It is crucial to comprehend the objectives and the expenses to attain positive results. The cost, which infrastructure and operational complexity can gauge, must be managed individually. The practicality and advantages of employing cloud and edge computing for digital twins are significant academic subjects. There is also a need for additional research and optimization to address such systems' real-time requirements and data management issues, enabling their application in edge computing.

Keywords: Digital Twins. MQTT. Apache Kafka. Cloud Computing. Edge Computing.

Projetando e Implementando Gêmeos Digitais com Computação em Nuvem e de Borda: Desafios e Oportunidades

RESUMO

Os gêmeos digitais são modelos interconectados de um sistema físico e um virtual que interagem em tempo real. O componente virtual é um software que espelha as ações da contraparte física, identifica problemas e os resolve de forma autônoma sem intervenção humana. O advento da Internet das Coisas e de dispositivos sensoriais acessíveis deu origem ao gêmeo digital contemporâneo, que requer a capacidade de controlar suas contrapartes físicas, necessitando de uma conexão confiável. Esta pesquisa investiga a relação entre sistemas de aquisição de dados e o processamento de eventos no contexto de gêmeos digitais utilizando computação em nuvem e de borda. O principal desafio desta pesquisa é criar métodos para aprimorar o desempenho de sistemas com uso intensivo de dados com entrada de dados crescente, respeitando as restrições de tempo, custo e complexidade. Este trabalho apresenta um estudo sobre gêmeos digitais viabilizado pela computação em nuvem e de borda, experimentando as necessidades de comunicação em tempo real de um sistema de gêmeos digitais implantado em um ambiente multicomponente em um cluster na nuvem e na borda. Este trabalho amplia o escopo para projetar sistemas de gêmeos digitais para operar em escala, realizando experimentos usando MQTT e Apache Kafka para identificar possíveis problemas para dispositivos de nuvem e de borda. Os experimentos empregaram uma configuração de rede usando um cluster Kubernetes, diferentemente das abordagens de simulação anteriores que utilizavam apenas máquinas virtuais, para obter resultados mais autênticos e precisos. Os experimentos revelaram possíveis problemas nas execuções dos cenários devido ao fato de os corretores exigirem mais recursos do que os dispositivos de borda de baixa especificação poderiam oferecer e, ao mesmo tempo, discutindo os compromissos e as restrições do uso do Kafka para gêmeos digitais, principalmente em cenários de baixa especificação. A introdução de uma camada de computação adicional em um aplicativo com uso intenso de dados pode sobrecarregar o hardware disponível para o sistema de gêmeos digitais. A configuração do Kafka também é uma tarefa complexa para gerenciar o particionamento de tópicos, a aquisição de dados e os fatores de replicação. É fundamental compreender os objetivos e as despesas para obter resultados positivos. O custo, que pode ser medido pela infraestrutura e pela complexidade operacional, deve ser gerenciado individualmente. A praticidade e as van-

tagens de empregar a computação em nuvem e de ponta para gêmeos digitais são assuntos acadêmicos importantes. Há também a necessidade de mais pesquisas e otimização para atender aos requisitos de tempo real e aos problemas de gerenciamento de dados desses sistemas, permitindo sua aplicação na computação de ponta.

Palavras-chave: Gêmeos Digitais. MQTT. Apache Kafka. Computação em nuvem. Computação em borda.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
AR	Augmented Reality
AWS	Amazon Web Services
BIM	Building Information Modeling
CapEx	Capital Expenditure
CoAP	Constrained Application Protocol
CPS	Cyber-Physical System
CPU	Central Processing Unit
DT	Digital Twin
FINEP	Financier of Studies and Projects
GB	Gigabyte
GiB	Gibibyte
GIS	Geographic Information Systems
HTTP	Hypertext Transfer Protocol
ICS	Industrial Control Systems
IIoT	Industrial Internet of Things
IoT	Internet of Things
JSON	JavaScript Object Notation
KRaft	Apache Kafka Raft
MQTT	Message Queuing Telemetry Transport
NFV	Network Functions Virtualization
OpEx	Operational Expenditure

OWL	Ontology Web Language
PaaS	Platform-as-a-Service
QoS	Quality of Service
RAM	Random Access Memory
RCN	Norwegian Research Council
REST	Representational State Transfer
RTOS	Real Time Operating System
SaaS	Software-as-a-Service
SDN	Software Defined Networking
SWRL	Semantic Web Rule Language
TiB	Tibibyte
VR	Virtual Reality
V2X	Vehicle-to-everything

CONTENTS

1 INTRODUCTION

Digital twins were introduced in 2002 as a two-space mirrored and interconnected model of a system, one real and one virtual, where each part reflects its partner, at the same time that they communicate in real-time (GRIEVES, 2016). The virtual part of a digital twin could be defined as a computer program built with the capacity to understand and generate simulations about the behavior of the real counterpart and how it will affect the data captured via sensors, capturing its characteristics and being capable of revealing the inner works of a system and diagnosing unintended problems of it, proactively acting to fix problems without the need for human intervention, basing its decisions on the acquired knowledge from the system (BOSCHERT; ROSEN, 2016).

Low-cost sensory devices, accessible due to the expansion of the Internet of Things (IoT), are one of the primary enabling factors of the modern digital twin, being a solely digital implementation that only requires a definition of assets and data transmission, without an actual duplication of the real system (TAO; ZHANG; NEE, 2019; TAO; SUI *et al.*, 2018; QI *et al.*, 2021). Using different sensors enables digital twins to collect data in real time and transmit it to a server in the cloud for analysis using big data and Artificial Intelligence (AI), which enhances the accuracy of the physical model simulations. In the physical world, sensors capture performance, behavior, and interactions, which are then used to provide feedback through the system's actuators. In the virtual world, the cloud generates corresponding virtual objects that visualize and understand the object's structure and simulate its use. Traditionally, virtual and physical objects' construction, analysis, and upgrade have been separate and need more unified integrated analysis. Digital twins, as a product framework design, can be used to integrate sensor data, knowledge, and actuators of the physical objects that compose the system to enable a data-driven system capable of in-depth simulations of the real system, knowing how to act based on past behavior, effectively fusing both system realities into an automated and autonomous one.

Digital twin research is a highly relevant research topic involving industry and academia. To enable recommendation of actions, digital twins require the ability to order physical products, with an established and secure real-time connection between the physical world and virtual reality (ZHANG *et al.*, 2022). The expansion of sensing technologies and the recent rise of AI enable even more seamless and integrated applications, which is evident with Virtual Reality (VR) and Augmented Reality (AR). In VR, AI-enabled sensing technologies can create more realistic simulations by accurately tracking

and responding to user movements and actions. Similarly, these technologies can be used in AR applications to overlay digital information in the real world, enabling users to merge the digital and physical worlds.

Real digital twins rely not only on optimizing data acquisition from its system components but also on organized processing and storage of data, considering its big data requirements of large datasets, both from an incoming live feed and from ever-growing historical data (KNEBEL, 2020). This study explores the correlation between data acquisition systems and event processing in digital twins - dynamic real-time systems that generate copious amounts of data and necessitate continuous processing. Scaling such systems can pose a challenge since outsourcing computing to the cloud may compromise their real-time requirements. The crux is devising strategies to optimize compute-intensive systems with growing data acquisition while adhering to timing constraints, budget limitations, and complexity considerations.

To better understand the behavior of data acquisition, processing, and storage systems for digital twins running on real cloud end edge deployments, a series of experiments were conducted using a combination of Raspberry Pi single-board computers, small form factor desktop computers, and server-grade machines. Unlike previous simulations, which relied solely on virtual machines (KNEBEL, 2020; BARROS, 2022), these experiments used a genuine network setup utilizing a Kubernetes cluster, resulting in more accurate and realistic outcomes that a proper implementation would face. Despite the valuable insights gained from the simulations, those experiments fell short of accurately replicating real-life scenarios. This work will present a study on cloud and edge computing-enabled digital twins, experimenting with the real-time communication requirements of a digital twin system implementation deployed to a multi-component cloud and edge cluster environment. This work aims to expand upon existing research in the area and aim towards how digital twin systems can be designed to operate at scale, executing multiple experiments, focusing on scaling multiple sources of data and processing using MQTT (Message Queuing Telemetry Transport) and Apache Kafka, and validating potential problematic scenarios for cloud and edge devices. The experiments present potential bottlenecks in the scenario executions caused by the brokers requiring more resources than were available to the low specification edge devices and discuss the viability of using event-processing and multi-broker systems with edge devices, considering features of the systems used and providing potential solutions.

The remainder of this work is organized as follows: Chapter 2 introduces essential

subjects for the understanding of this work, including definitions of cloud computing, real-time systems, and digital twins; Chapter 3 details other important work related to what was researched and implemented; Chapter 4 presents the context behind the work being presented and proposes the framework to solve the original problem; Chapter 5 presents different experiments on said framework and follow-up discussions, discussing opportunities, challenges and where future implementations can improve on this work; and Chapter 7 ends with a conclusion summarizing the essential aspects of the presented work. Annexed works include other papers written that are referenced and an expanded summary in Portuguese.

2 BACKGROUND

This chapter briefly introduces some essential subjects to this dissertation, explaining the concepts of Cloud, Edge and Fog Computing, Real-time Systems, and Digital Twins, and discussing how they relate to each other and their importance to the work presented.

2.1 Cloud, Edge and Fog Computing

Distributed computing can process the vast amounts of data generated by digital twins, allowing for real-time analysis and monitoring of physical assets and systems, and this can involve using distributed databases, cloud computing, edge computing, and other technologies to distribute the workload and processing power needed to handle the data generated by the digital twin. Cloud computing, edge computing, and fog computing are all related to distributed computing and involve processing and storing data outside a traditional centralized data center. However, they differ regarding where the processing and storage occur and the applications they are best suited.

Cloud computing refers to delivering different computing services, including servers, storage, databases, networking, software, analytics, and intelligence, over the Internet. The combination of these elements is what we call “the cloud”. Data processing is decentralized from the user and dispatched to a remote data center without the hardware limitations of the machine used by the user. Cloud computing is beneficial for applications that require high scalability, availability, and elasticity, as well as those that involve big data processing and analysis, due to the significantly larger pool of resources that a complete data center can dedicate. Cloud computing offers a convenient and on-demand approach to accessing shared computing resources (MELL; GRANCE *et al.*, 2011). Cloud computing enables geographical distribution by transferring data processing and storage responsibilities to remote data centers, increasing system redundancy and reliability (MARINESCU, 2017; DENG *et al.*, 2010). Another important factor in cloud usage is cost. The user can ask cloud providers for resources in a pay-as-you-go model, where the pricing model is based on the usage of resources, which may be variable, allowing users to scale up or down as requirements change without having to acquire or set up any hardware themselves. Overall, cloud computing costs are highly variable but can provide significant cost savings compared to a traditional on-premises infrastructure

since the cloud provider will be responsible for all upkeep on hardware, software, and personnel maintenance.

All these benefits are essential and are a substantial contributing factor in implementing many modern applications we use today, but cloud computing also has disadvantages. Security in data privacy is one of them since storing large amounts of sensitive data on the Internet also allows for these data points to be maliciously breached. Other important issues to note are the reliance on access to the Internet, increased application complexity, and possible hidden additional costs if the application needs to be optimized correctly or if the pricing model needs to be adequately understood. Architecturally speaking, cloud computing performance is strictly affected by the additional network latency inserted in the system to offset computing to its remote data centers. Real-time applications with strict timing rules bound to the application logic are examples where cloud usage may not be viable, or at least part of the application must be split between what can be deployed in the cloud and what must be executed on-premises.

Edge computing can help address some of these challenges. By bringing processing data closer to the data sources, edge computing reduces the overall transmission time over the network. If the computation can be executed in less time than it takes for a cloud computing approach, it improves the application's overall performance. In other words, programs with real-time or near-real-time processing, such as IoT devices, autonomous vehicles, and industrial control systems (based or not in digital twins), instead of sending data to be processed in a data center multiple hops away, execute them as closer to the local network as possible, which avoids the increased physical latency of communication. Edge computing brings processing closer to the source, reducing cloud traffic and service latency. By enabling computation closer to where it is required, edge computing significantly enhances response times (CHEN *et al.*, 2019).

Fog computing distinguishes itself from the cloud by its decentralized nature. It expands cloud computing into the physical world and its many connected devices, ensuring data availability precisely when and where it is needed (ATLAM; WALTERS; WILLS, 2018). In contrast, edge computing represents the closest possible deployment point to the data source. However, it primarily comprises devices that need complete control from the cloud provider, presenting significant challenges regarding system reliability and availability. Fog computing, on the other hand, enables latency reduction while still operating in a more controlled environment. Fog computing is similar to edge computing in that it involves processing and storing data at or near the network's edge. However, fog

computing typically involves a more distributed architecture, where computing resources span multiple nodes or devices, effectively being a middle-ground approach between edge and cloud, offering more computing resources to the user while reducing the effects of increased latency in data transmission.

2.2 Real-time, Cyber-Physical and Digital Twins

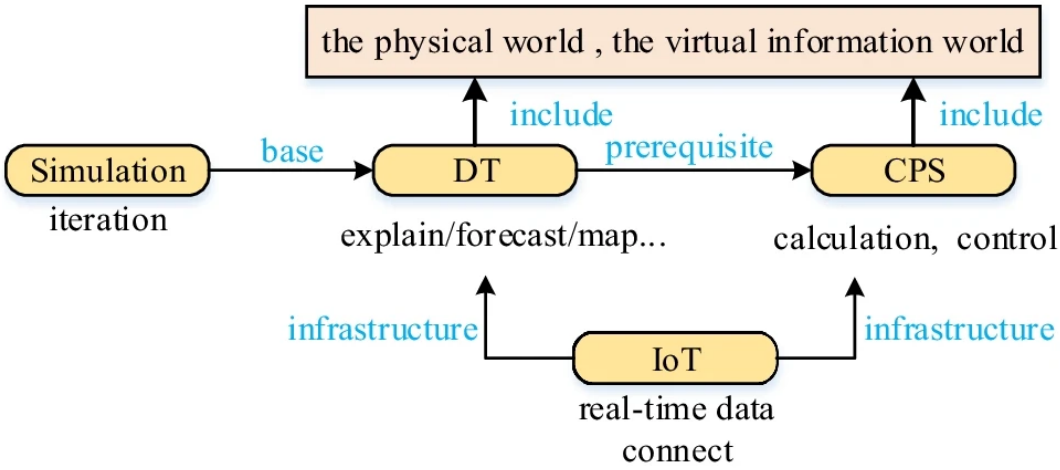
Real-time systems are computer systems that must respond within a specified period in applications where timing is crucial, such as control systems, robotics, and telecommunications. Tasks in real-time systems have strict timing requirements, and failure to meet these requirements can have catastrophic consequences. Real-time systems typically operate in a closed-loop manner, continuously monitoring and adjusting their actions based on feedback from sensors or external inputs.

There are two primary categories of real-time systems: hard real-time systems and soft real-time systems. Hard real-time systems have strict timing requirements and must respond within a specific timeframe, while soft real-time systems have more flexible timing requirements and can tolerate some degree of delay. Real-time systems are widely used in various applications, including aviation, healthcare, automotive, and industrial control systems. These systems rely on Real-Time Operating Systems (RTOS) to manage the timing requirements and ensure proper system functionality.

Despite the variations in how scholars and practitioners define the digital twins, some common characteristics can be identified: (1) digital assets are used to mirror reality; (2) the digital twin can be refined or modified based on the bidirectional feedback between physical and virtual assets; and (3) evolution and high-fidelity models are crucial for the implementation of digital twins. These characteristics resemble the components of simulation, Cyber-Physical Systems (CPS), and IoT, as exposed in Fig. 2.1.

Digital twins go beyond being a mere digital model or interface of a physical object. They are designed to receive input from various sensors connected to their real-world counterparts. With this data, digital twins can simulate the physical object's behavior in real time, providing valuable insights into its performance and identifying potential issues resulting from incorrect or invalid behaviors that might occur in the system. Digital twins can also be utilized during a system's design stage. In that scenario, the digital twin serves as a prototype of the real system, enabling testing and optimization before any physical system is constructed, which ultimately helps to reduce project costs.

Figure 2.1 – The relationship between DT, CPS, IoT, and Simulation.



Source: (XIONG; WANG, 2022), p. 3

A digital twin is a complex system that integrates a digital representation or interface of a physical object with various sensors that monitor its real-world counterpart. The data collected by the sensors enables the digital twin to simulate the physical object in real time and to analyze its performance and potential problems. A digital twin aims to establish a dynamic relationship between the digital and physical systems, where they interact in a continuous feedback loop. The real system provides data to the digital system, and the digital system provides suggestions to the real system, enhancing its processes autonomously. A digital twin requires both real-time and simulation aspects to function correctly. Without the simulation aspects, it becomes a *digital model*, a static representation of the physical system. Without the real-time aspects, it becomes a *digital shadow*, a simple visualization of the physical system (KRITZINGER *et al.*, 2018).

The proliferation of low-cost sensor devices enabled by IoT is a significant driver for developing digital twins. Digital twins are an entirely digital and easy-to-implement solution that does not need a physical duplication of the real system but only the definition of the system components and their data input. Implementing cloud-based digital twins can improve the integration between parts and end-user access, accelerating product development and manufacturing. The software platform components can be expanded and scaled on-demand (KNEBEL, 2020).

Utilizing a cloud-based microservices digital twin allows easy integration between its parts and convenient accessibility for end-users, leading to faster product development and manufacturing. The platform's components can be scaled and expanded on-demand with independent development between modules. Any improvements to the indi-

vidual microservices can be easily integrated with minimal changes to the entire system (JOSEPH; CHANDRASEKARAN, 2019). The digital nature of a digital twin system's virtual component allows it to be decentralized, physically separated from its real-world counterpart (CARDIN, 2019). Integrating fog and edge computing with IoT and digital twins enables the creation of a mirror image of physical and virtual spaces while meeting the real-time system's timing requirements. However, managing, processing, and storing data in a highly distributed environment still poses a challenge for digital twins (DIZDAREVIĆ *et al.*, 2019; KHAN *et al.*, 2020). More details about the implementation of digital twins, specifically in the context of cloud and the oil and gas industries, are included in the paper "*A Study on Cloud and Edge Computing for the Implementation of Digital Twins in the Oil & Gas Industries*", annexed in preprint form in Appendix B.

3 RELATED WORK

Digital Twin cloud services refer to cloud-based platforms that enable the creation, deployment, and management of digital twins. Cloud services provide a range of features and functionalities, including data ingestion, storage, analysis, and visualization. They also offer tools for creating and managing digital twin models and APIs for integrating with other applications and systems. Some popular digital twin cloud services are discussed in this section. This chapter details other important work related to what was researched and implemented.

3.1 By the industry

The proposed research solution does not aim to rival established industry tools. Instead, it is a valuable platform for studying and analyzing the field's current state. The software tools utilized in this research are built on open-source frameworks and libraries widely recognized and employed by the research and industry communities. These tools were carefully selected to facilitate the implementation, evaluation, and comparison of various techniques and methods for constructing digital twins while also enabling the verification and replication of the results presented. The following sections present different cloud platforms for digital twins, studied to identify gaps and limitations of existing tools to propose new solutions or improvements that can address them.

3.1.1 Microsoft Azure Digital Twins

Azure Digital Twin, developed by Microsoft as a Platform-as-a-Service (PaaS) offering, is a digital twin architecture used to create and manage digital twin models of physical environments like cities, buildings, and factories. This system allows users to simulate and visualize their physical environments, monitor asset performance in real-time, and leverage predictive analytics to optimize operations and reduce downtime (MICROSOFT, 2023a). Users can also design a digital twin architecture that represents IoT devices, which can be connected to Azure IoT Hub device twins to send and receive live data or to REST APIs or other connectors inputting live data. While IoT Hub device twins are maintained by the IoT hub for each IoT device connected to it, digital twins in Azure

Digital Twins can represent anything defined by digital models and instantiated within the system. Within Azure Digital Twins, users define digital entities that represent the people, places, and things in their physical environment using custom twin types called models. These model definitions are like a specialized vocabulary to describe the user's business. For instance, a building management solution might define a Building type, a Floor type, and an Elevator type. Models are defined in a JSON-like language called Digital Twins Definition Language (DTDL), which describes the types of entities according to their state properties, telemetry events, commands, components, and relationships. DTDL is a customized JSON-LD-based model similar to OWL ontologies. Users can design their own model sets from scratch or begin with a pre-existing set of DTDL industry ontologies based on a shared vocabulary for their industry. External client applications can connect to the twin instance to manage/query the model graph, and external compute services can leverage the twin instances to perform analysis and generate insights. Outputs can then be forwarded to other compute and storage PaaS services in the Azure cloud for storage and analysis of data accumulated over time by the digital twin (MICROSOFT, 2023b).

3.1.2 AWS IoT TwinMaker and IoT SiteWise

Amazon offers two different solutions in the digital twin perspective: IoT TwinMaker and IoT SiteWise. AWS IoT TwinMaker is a general-purpose service that helps developers create digital twins of real-world systems such as buildings, factories, industrial equipment, and production lines. With AWS IoT TwinMaker, developers can use existing data from multiple sources, such as IoT devices, video cameras, and enterprise applications, without needing to reingest or move the data to another location, using the broad offering of IIoT cloud services from AWS. It can compose immersive 3D scenes of the physical systems and overlay real-world data on them to get a holistic view of their operations via the Scene composer, where the user can submit previously built 3D/CAD models and place multiple models in a single scene, creating a virtual representation of their operations (OKEKE, 2022). AWS IoT TwinMaker is a service that allows customers to create and manage digital twins of their physical systems. Customers can use AWS IoT TwinMaker to import pre-existing 3D models of their physical system components to develop a digital twin. Then, they can overlay the 3D models with knowledge graph data that captures the relationships and properties of the physical system. This way, they can produce a realistic and dynamic digital twin that can be used for various purposes such as

monitoring, testing, or optimization (CAN; TURKMEN, 2023).

AWS IoT SiteWise, on the other hand, is directed directly at collecting and organizing data from industrial equipment at scale, using time series data stores. Users can utilize SiteWise assets with TwinMaker, automatically syncing changes in models between both services [1]. It can then be used to define the components and the data collection used in TwinMaker.

3.1.3 Siemens MindSphere and Insight Hub

Siemens MindSphere was a cloud-based IoT platform with a digital twin solution for industrial use cases, connecting assets and data to explore insights. It has been integrated into the Siemens Insights Hub as one of its services, aiming to drive smart manufacturing through IIoT. It combines digital twin technology with product lifecycle management systems to create a closed-loop digital twin, which can help companies optimize their product and manufacturing process. (SIEMENS, 2023b,a). Key features include:

- Model-based approach to creating and managing digital twins;
- Integration with other Siemens industrial software and services;
- Collect and analyze key data during operation;
- Real-time monitoring and analytics of industrial assets and processes;
- Tools for predictive maintenance and optimization to optimize performance and reduce downtime;
- Improved production visibility;
- Optimizing energy consumption of physical assets.

3.1.4 IBM Watson IoT Platform

The IBM Watson IoT Platform provides a digital twin solution for IoT devices and systems, which allows users to create virtual representations of their physical assets and monitor them in real-time. It includes data collection, analysis, visualization tools, machine learning capabilities for predictive maintenance and optimization, and integra-

[1] Asset synchronization with AWS IoT SiteWise documentation can be accessed at `https://docs.aws.amazon.com/iot-twinmaker/latest/guide/tm-sw-asset-sync.html`

tion with other IBM solutions such as Watson Studio for advanced analytics and machine learning (IBM, 2023). Key features include:

- Integration with IBM Watson services for advanced analytics and automation;
- Machine learning capabilities for predictive maintenance and optimization;
- Real-time monitoring and analytics of IoT assets and systems;
- Visualization tools for creating custom dashboards and reports.

3.1.5 Bentley iTwin

Bentley Systems is a leading global provider of software solutions for infrastructure engineering, construction, and operations. In 2020, Bentley announced the launch of their new digital twin cloud service, called iTwin, providing a comprehensive platform for creating, visualizing, and analyzing digital twins of infrastructure assets, such as buildings, roads, bridges, and utilities (BENTLEY, 2023; RUBENSTONE, 2023). Key features include:

- Automated change detection and analysis to identify and track changes in the physical asset and its digital twin over time;
- Simulation and what-if analysis to test different scenarios and optimize asset performance;
- Machine learning and AI-powered analytics to identify patterns and insights in large amounts of data;
- Tools for creating and configuring digital twin models;
- Real-time monitoring and analytics of digital twin models;
- Visualization tools for creating custom reports;
- Integration with other Bentley software solutions;
- Support for multiple data sources and standards, including BIM (Building Information Modeling) and GIS (Geographic Information Systems).

3.2 By academia

Digital twins in real-time scenarios have been a hot academic topic for years. Some factors that can be used to compare different digital twin platforms are:

- The level of integration with IoT devices, sensors, and data sources;
- The type and quality of simulation, visualization, and analytics tools;
- The support for AI and ML techniques to enhance the intelligence and learning of digital twins;
- The scalability, reliability, and security of the platform architecture and infrastructure;
- The platform's ease of use, customization, and interoperability.

Shankar *et al.* (2022) proposes a knowledge-based digital twin prototype for the oil and gas upstream process, using a generalized IoT microservice stack and schema-based ontologies. The paper presents a solution using open-source technologies and open standards: Apache Jena, Ontology Web Language (OWL), Semantic Web Rule Language (SWRL), MQTT, and MongoDB. The model implemented represents the relationship between the various assets and values obtained from logs, sensors, and configuration information. The paper demonstrates the testing and results of the prototype using real data from an oil well and shows a use case of using the acquired knowledge base and sensor data to perform a proactive site visit scheduling before equipment performance is affected and degrading the quality of the extracted product and also marking equipment which shows signs of a possible future failure as non-compliant so that administrators are notified and can proactively restore regular operation (SHANKAR *et al.*, 2022)

In a previous work, I Knebel, Trevisan *et al.* (2023) reviewed the literature on digital twins, cloud and edge computing, and the Oil and Gas industry, using a systematic approach to classify relevant articles and identify these technologies' current state and challenges in this context. The review states that the adoption of cloud and edge in the Oil and Gas industry was late compared to other industries, primarily due to data security and privacy concerns. The paper summarizes the key concepts and features of digital twins and cloud computing, mentioning cloud platforms and services to support the development of digital twins. It also discusses using different cloud models and highlights security as a significant concern when using cloud computing. Cloud-based extensive data management and processing, like event streaming, in-memory computation, storage, and visualization, are also significant topics in the area (KNEBEL; TREVISAN *et al.*, 2023).

C2PS is another architecture proposed as an ontology-based architecture for DT, leveraging machine learning capabilities using the Bayesian belief network. For C2PS,

network connections are omnipresent, and every physical *thing*, physical equipment and devices, are automatically accompanied by a cloud-hosted digital twin with a direct connection. Whenever the physical world changes, a physical sensor tries to update the current status to its digital twin representative, with data stored locally and in the cloud. C2PS presents the idea of an "intelligent service layer" decoupled from the cyber entity, containing sets of services that include the relations and ontologies of the *things*, being able to communicate with the other devices, cyber or physical. The C2PS architecture reference model works as a template so that cross-domain integrations can be seamless through a common language formed by a meta ontology of the system so that data can be easily shared between domains to construct smart cities (ALAM; EL SADDIK, 2017).

López (2021) presents a case study of an actual industrial flotation procession for a mining plant, capturing data, storing, and processing data from sensors and devices to feed an artificial neural network that predicts the rate of silica concentration and provide information to engineers on-site, to act in a preventative and optimized way. High performance in data processing is essential when merging the data streams and the information modeling of the DT due to the low latency required for critical applications. The authors propose utilizing a broker topology using Apache Kafka to send the data streams to be processed asynchronously. They can elastically scale across multiple servers and use replication to mitigate potential data loss in case of failure while integrating with other Apache data processing systems. The article concludes that the platform is applicable for developing DT applications in various areas, such as manufacturing, production, or mining, and that it meets the requirements of scalability, robustness, reliability, real-time performance, and data-driven decision-making (LÓPEZ, 2021).

Kafka is widely used in several digital twin implementations. Correa (2022) explored a microservice-based approach towards Oil & Gas digital twins, focused on the flow assurance process of an oil well, breaking down the functional requirements into more minor services. The implementation does not stress the communication to see how it would scale in a non-simulated scenario. Problems presented include the importance of data security for these types of implementations. The cloud deployment process is also significant, suggesting that using a public cloud platform could help in security and compliance, providing more control over the data being processed (CORREA, 2022).

Alonso, Locci e Reforgiato Recupero (2024) present an infrastructure digital twin implementation focused on bringing AR elements to a working office, monitoring temperature, humidity, and air quality to measure personnel well-being, and expanding digital

integration. Integrating multiple sensors and actuators can improve digital immersion, but integrating sensors from different vendors using incompatible data standards results in complications when merging all components (ALONSO; LOCCI; REFORGIATO RECUPERO, 2024).

Fennell (2022) uses Kafka to test the acquisition of real-time sensor data from a motorway (GPS data, traffic cameras) and uses it on a digital twin, testing the effects of compression, batching, and different hardware configurations through simulations. A single broker design was found to have the highest performance in terms of latency, but that results in a single point of failure, so it used a three-broker configuration, sending compressed data and replicating across the brokers to improve reliability (FENNELL, 2022). A problem with this solution, in terms of implementation of edge microservices, is the additional required hardware: Kafka is very resource intensive, with suggestions of at least 64 GB of RAM and 24 CPU cores for operating at a high level (CONFLUENT, 2024a). Creating at least three broker nodes and all the other components needed for the stack to function makes this implementation unsuitable to deploy at the edge, requiring a centralized cloud.

3.2.1 Digital Twin Networks

For the Software-based and Programmable Networks course, a research paper (in Portuguese, Appendix C) was written featuring systematic research combining computer networks, SDN (Software Defined Networking), NFV (Network Function Virtualization), and digital twins. Its main goal is to define the state of the art of digital twin research in the context of programmable networks. It also looks at the possible contributions that SDN and NFV can bring toward implementing digital twins. From the 200 most relevant results from two queries made in Google Scholar, 67 papers were analyzed and included in the final results. The article list generated nine different research topics, presented in Table 3.1, as a summary of the contributions of the three key topics of the research and how they interact with each other for computer networks. The final results of this work (in Portuguese) are attached in Appendix C.

Topics brought up by papers in *5G, 6G, and Wireless Systems* include using a centralized controller coupled with a digital twin performing network monitoring, maintenance, and diagnostics. Network slicing is another critical concept, performing network virtualization for resource allocation to interested parties while having autonomous man-

Table 3.1 – Digital Twin Networks Paper Classification

Categories	**Total**
5G, 6G, and Wireless Systems	27
Anomaly Detection	2
VNF Scaling	1
Network Digital Twins	16
Internet of Things	17
P4	2
Optical Networks	3
Vehicular Networks	4
Security	9

agement and orchestration to guarantee application QoS (Quality of Service). Data decoupling, proactive analysis, and resource optimization by adding intelligence to the network would also reduce CapEx (Capital Expenditure) and OpEx (Operation Expenditure) in network management.

For the *Internet of Things*, the growing number of devices, combined with data generation in a virtualized environment, can be managed by digital twins, creating dynamic forwarding rules in the network to guarantee real-time conditions for the Industrial Internet of Things. Examples include the precision and reaction time required by industrial robots operated over long distances. Migration of these virtualized devices towards the edge can also improve the usability of mobile devices, transferring the state of its coupled digital twin in the edge layers without loss of computation. Time-sensitive networking implemented with digital twins applies strict real-time rules to computer networks, optimizing routing and scheduling and providing a global overview of the network with constant updates via telemetry data.

One of the selected papers includes network virtualization using P4, creating digital twin-distributed networks connected to a shared backbone. The results of the P4 implementation are then compared to other non-P4 related works, without a clear conclusion of the best case scenarios, but allowing for significant scalability without adding physical switches (LARSSON, 2021).

For *Vehicular Networks*, concerns include how to enable a decentralized intelligent network, suggesting the usage of the SDN controller to introduce additional computational capacities to the network, having a global view of the network, and adaptive capability towards changes in the network. Autonomous cars are a heavily linked subject, and introducing digital twins helps in terms of resilience, safety, and fault tolerance, both mechanical and from possible cyberattacks, while helping with route planning, coordina-

tion of virtualized vehicles, and task offloading. Another important feature is providing low latency services in an environment with high mobility, using virtualization and migrating digital vehicles in the network towards the closest edge nodes. Using digital twins for zero-touch networking is another possibility that allows networks to have automated updating and provisioning capacity without human intervention.

Digitalization of assets opens breaches to attacks against physical systems that were previously unreachable, thus creating the need for increased protection since attacks would cause direct effects on the physical world. SDNs can create more resilient systems by creating redundant paths and cyberattack mitigation strategies. Using virtualized ICS (Industrial Control Systems) is another possibility, introducing a plane of security inside the softwarized network, smart operations, and maintenance with automated and intelligent policy modification handled by the digital twin. Network digital twins, implemented with the help of SDN and NFV technologies, assist in problem-solving, predicting the future states of the network, and providing an automated increase in reliability.

4 PROPOSAL

This chapter presents the context behind this work, comments on previous works, and defines the Digital Twin architecture of the proposed solution.

4.1 PeTWIN project

Aimed at digitalizing the production process in digital twins that allow simulating all the production conditions of an oil plant, the PeTWIN - Digital Twins for Production Optimization and Management project[1] is a partnership between the Research Group Computing Systems for Oil E&P of INF-UFRGS and the Sirius Lab of the University of Oslo, made possible through the Financier of Studies and Projects (FINEP) and the Norwegian Research Council (RCN), co-financed by the companies Petrobras, Shell, Total, CNPC and CNOOC, which make up the Libra consortium in Brazil, and the Norwegian Shell and Equinor.

The PeTWIN project provides scientific insight and best practices for implementing sustainable, usable, and maintainable digital twins for field management. Simply put, this digital twin is a data repository associated with a simulator that monitors the operation of an oil production facility, enabling online data analysis and predictive assessment. PeTWIN will address the computer science challenges presented by digital twins through applying and developing techniques for capturing and processing physical data in real-time, conceptually modeling this data under an integrated view, and analyzing data. While digital twins are a well-known technology for industrial equipment, applying a digital twin for oil production assets is relatively new and innovative in integrated data analysis.

The main objective of this group is to define a conceptual system architecture for the next generation of digital twins for the oil production field. Initially, we understand this architecture will be inspired by the five-dimension DT model proposed by Tao et al (TAO; ZHANG; LIU *et al.*, 2018), illustrated by Fig. 4.1. These dimensions include:

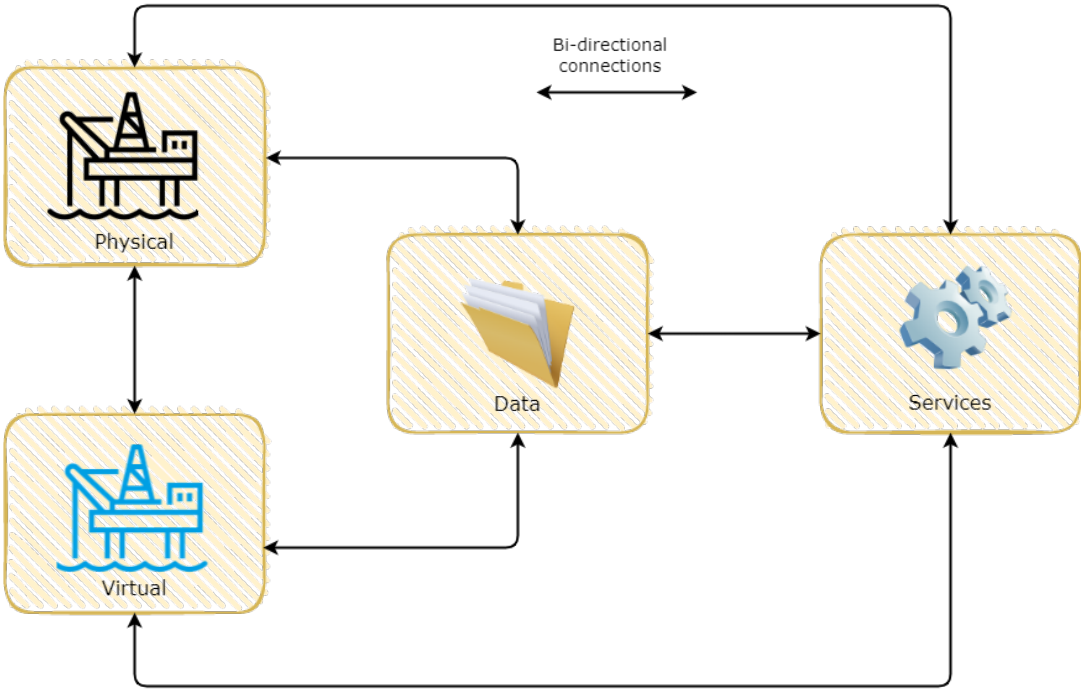
1. a Physical entity with various functional subsystems and sensory devices;
2. a Virtual high-fidelity digital model of the Physical entity;
3. a set of Services that optimize operations of the Physical element and calibrate

[1] More information can be found at the project's homepage: `https://petwin.org`

parameters of its Virtual model;

4. data acquired from both the Physical and Virtual elements, domain knowledge, as well as any fused, transformed, and enriched information extracted from the raw data;
5. bidirectional Connections, representing communications and flow of encoded information among the other four dimensions.

Figure 4.1 – Summarization of the five-dimension digital twin model



Source:
https://petwin.org/system-and-deployment-architecture/

In addition to outlining the theoretical system architecture, we also focused on the practical aspects of implementing and deploying the defined conceptual components. They utilize cloud-based technologies to meet the scalability, reliability, and quick response times demanded by contemporary Digital Twin systems.

4.2 Open Digital Twin

Motivated by this project, some work was developed prior to its start to study and better understand the state of the art in digital twin research. One problem identified in the literary research on this subject was finding open and easily replicable implementa-

tions. Due to this, the initial focus was on building an open architecture for digital twins, built from a collection of microservices and hosted in an organization on GitHub called Open Digital Twin[2], where the software is actively developed, and experimentation data is stored. This study generated the writing of multiple papers on digital twins before their link to the PeTWIN project.

During the 2020/1 academic semester, as a Special Student in the PPGC's "CMP158 - Real Time Systems" discipline, work was done linking the themes of digital twins with real-time systems, where it was observed that for digital twins with real-time requirements depend not only on the logical result of their computation, but also on their temporal limitations, and, to handle the large amount of data that must be stored and analyzed, digital twins rely on cloud-based architectures, which generates an additional latency factor in the computation performed, which can be mitigated by using edge and fog computing, reducing response times to comply with their time requirements (KNEBEL; WICKBOLDT; FREITAS, 2020).

Another parallel work was developed on the Open Digital Twin architecture, analyzing the deployment and communication limits via the use of the MQTT protocol, evaluating the use of the communication type Publisher-Subscriber in the context of digital twins, observing the limits of the broker that manages message delivery and examining its behavior in centralized and distributed situations, with more and less allocated resources (TREVISAN; KNEBEL; WICKBOLDT, 2020).

Connecting all the research done so far, another paper doing a bibliographic survey about digital twins, contextualization, their use cases, and established platforms, presenting the architecture developed by the Open Digital Twin and its motivations to be an open system and based on microservices, to be flexible and with reduced complexity for integration with third-party applications. The study achieved up to 64% reduction in average message transmission time for the digital twin when deploying on fog/edge computing nodes, compared to a cloud-only approach (KNEBEL, 2020).

Another paper proposed a scalable and application-aware data acquisition system for digital twins, testing the MQTT protocol and cluster mode. The paper recognizes the limitations of the MQTT protocol under high traffic conditions and explores alternatives to increase data flow while minimizing message loss and resource usage. The paper develops a "Cluster Manager" component that monitors the health of a cluster of MQTT brokers and adjusts the cluster composition according to the message flow. The paper conducts

[2]Available at `https://github.com/Open-Digital-Twin`.

experiments to determine the loss margins and latencies of sending critical messages, using different strategies to scale the cluster under representative workloads of digital twin applications. The paper also determines that using susceptible parameters to scale the system causes more nodes than necessary to be instantiated and resources wasted (TREVISAN; KNEBEL; WICKBOLDT; ABEL, 2022)

To finish, a significant task involved conducting a comprehensive literature review on digital twins, cloud and edge computing, and their application in the Oil and Gas industry. This review systematically categorized pertinent articles and pinpointed these technologies' present status and obstacles in this specific context. The study reveals that the Oil and Gas industry lagged behind other sectors in adopting cloud and edge technologies, primarily due to data security and privacy worries. The document offers an overview of the fundamental concepts and characteristics of digital twins and cloud computing, including cloud platforms and services that aid in creating digital twins. It further explores the utilization of various cloud models and underscores security as a paramount concern in using cloud computing (KNEBEL; TREVISAN *et al.*, 2023).

The Open Digital Twin organization continues to be a space used for prototype development for digital twin architectures in a project-agnostic way, without having direct ties, but being an open space for experimentation and development of prototype architectures for digital twins.

4.3 System Architecture

Evolving from the previous works, there are several obstacles to implementing digital twins in a way that is both scalable and sustainable. Latency is a significant issue with IoT communication, particularly in industrial settings. As adoption and implementation increase, the importance of low-latency services will become even more apparent (FERRARI *et al.*, 2017). Large-scale communication growth and increasing demand may limit technological innovation related to digital twins. Therefore, it is crucial to consider factors such as system scalability when implementing digital twins.

The proposed architecture is based on a stack of microservices. Unlike monolithic software, which is developed as a single, large codebase, this solution follows a microservices architectural approach. This approach allows the development of the DT as separate, self-contained, small pieces of software, each responsible for solving a specific task. This isolation of each process enables individual deployment of each application on-demand,

with each part working independently. In software changes, modifications introduced to a specific part of the system only affect that module, requiring updates only to that part. This feature of microservices-based architectures enables the scaling of services through a software deployment tool such as Docker or Kubernetes, selectively deploying only the necessary services, thereby reducing hardware resource waste.

Certain services within the architecture utilize third-party open-source software. One major challenge in designing a real-time, temporal data-generating system is selecting the appropriate storage engine for time-series data (DAMJANOVIC-BEHRENDT; BEHRENDT, 2019). This storage engine must be capable of querying and aggregating large volumes of sensor data obtained from multiple sources distributed across different geographic locations. This architecture relies on the MQTT protocol for communication between physical devices and the DT instance. The use of MQTT for telemetry in IoT devices has been established and tested in real-world scenarios and is currently employed in popular platforms such as Microsoft Azure IoT Hub (MICROSOFT, 2018), Amazon Web Services (AWS) IoT (BARR, 2015), and Facebook Messenger (ZHANG, 2011).

There is extensive documentation regarding the use of MQTT as a solution to these challenges within the context of IoT and digital twins (GRGIĆ; ŠPEH; HEĐI, 2016; ATMOKO; RIANTINI; HASIN, 2017; JACOBY; USLÄNDER, 2020). MQTT is designed to be a lightweight message exchange protocol that can be executed by devices with minimal computing power and connected over unreliable networks. It is widely used and standardized for IoT device usage. An MQTT architecture consists of two types of systems: clients and brokers. Brokers serve as intermediary servers with which clients communicate, exchanging data without direct client-to-client connection.

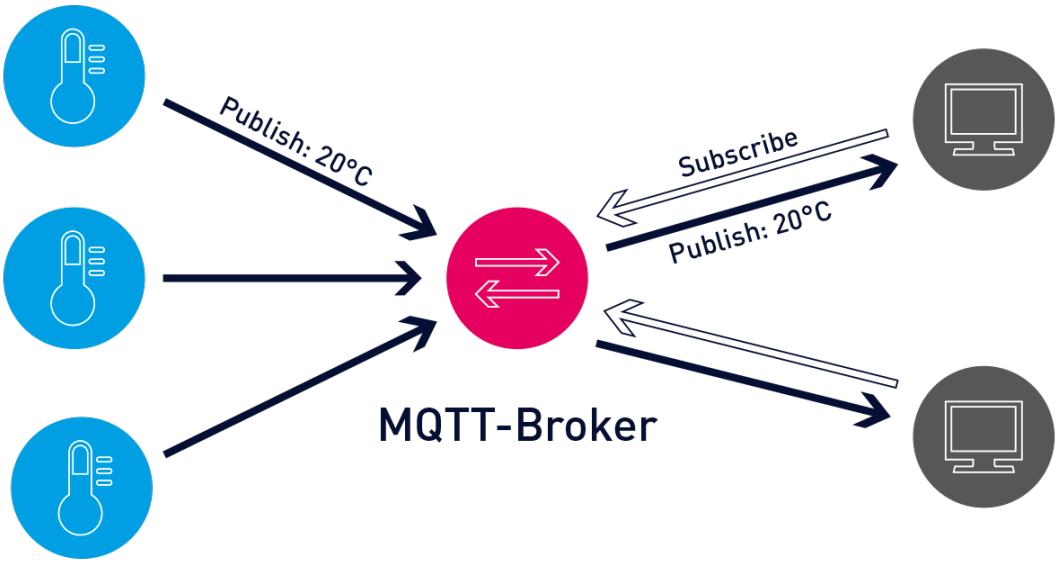
Despite being a lightweight protocol, MQTT messages can be configured to ensure a higher certainty of delivery, accomplished through setting the Quality of Service (QoS) (BANKS *et al.*, 2019). When the network is dependable yet restricted, utilizing QoS 0, which sends data without expecting an acknowledgment, may be the optimal option. However, QoS 1 and 2 offer greater message delivery assurance by allowing the client and broker to communicate and ensure the message has reached its intended destination. The differences between each level are:

- QoS 0: messages are delivered **only once** and not guaranteed to be delivered to the receiver. The message is sent from the publisher to the broker and immediately forwarded to the subscribers. The subscriber does not send any acknowledgment to the publisher or the broker.

- QoS 1: At this level, messages are guaranteed to be delivered **at least once**. The publisher sends the message to the broker, and the broker sends it to a list of subscribers. The broker will send an acknowledgment back to the publisher once the message has been received, which is resent if the publisher does not receive it after a certain period.
- QoS 2: At this level, messages are guaranteed to be delivered exactly once. The publisher sends the message to the broker, and the broker sends a message to the subscriber to acknowledge receipt. The subscriber then sends a message to the broker to confirm that the message was received. The broker sends an acknowledgment back to the publisher once it has received the confirmation from the subscriber.

Fig. 4.2 illustrates this type of communication, where temperature sensors transmit data using the Publish function, and other devices receive it using the Subscribe function. MQTT messages are published and identified within the broker using topics that denote the data source. The device within the established architecture and clients select which topics they want to become subscribers.

Figure 4.2 – Example of Publish/Subscribe communication, using MQTT.



Source: `https://www.paessler.com/it-explained/mqtt`

In building the architecture, a set of objects was defined to create a model of the Digital Twin (KNEBEL, 2020). These objects enable users to represent the physical environment as digital models and further enhance them by incorporating additional elements.

- User: responsible for creating, modifying, and updating their twin instances.

- Twin: the central structure of the instantiated DT, representing the digital version of the physical system and associated with all modeled objects.
- Element: a generic object used to define a part of the twin. It can represent any physical space or device, such as buildings or sensors, and is directly linked to a twin. Elements can also be associated with other elements to create more complex systems, depending on the system modeled. Note that elements are currently used as the location for data sources, but future versions could include additional information for physics simulations and the complete lifecycle of the physical element.
- Source: a generic component of an element used to define data entry points within the elements of a twin instance.

To overcome the issue of message loss during communication, some MQTT brokers offer the option of working in a clustered mode, where multiple brokers are connected and work together to achieve better performance, availability, and load balancing, which ensures a robust and fault-tolerant system with high processing power.

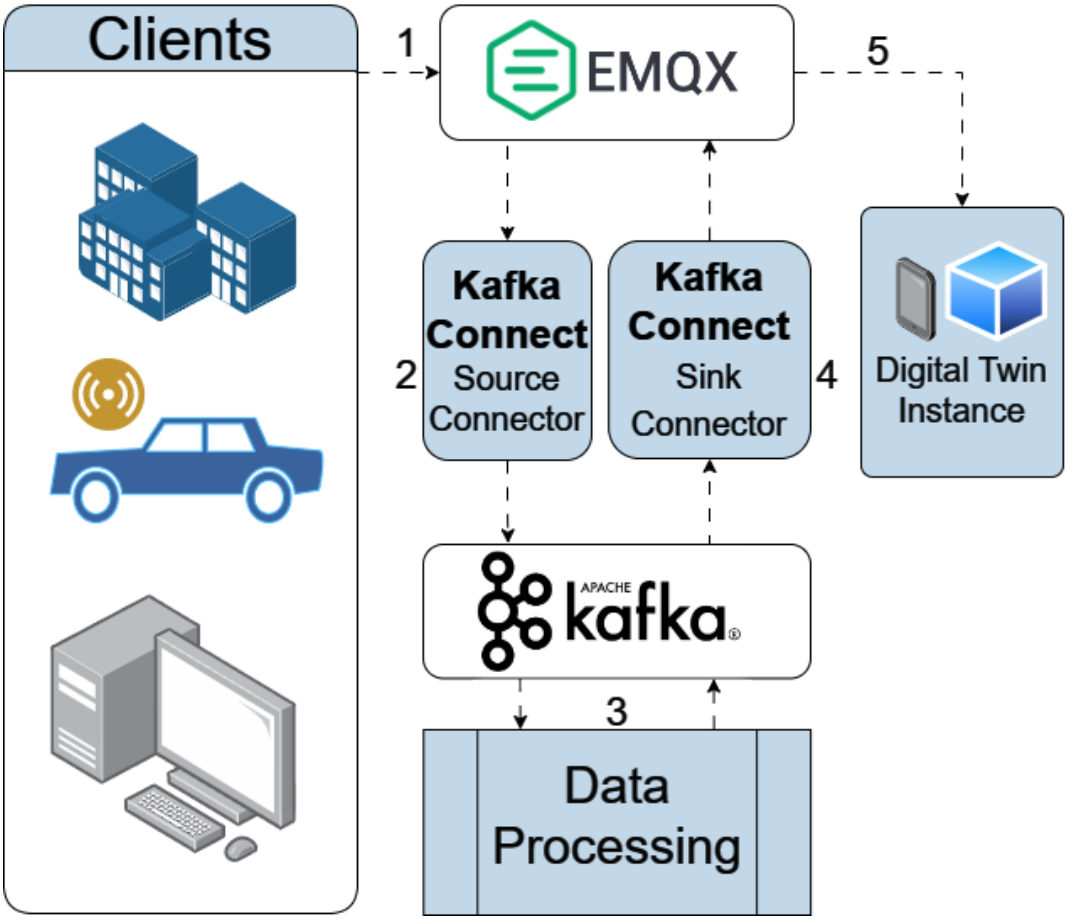
However, latency issues persist even after the cluster receives the data. When dealing with large amounts of data, such as digital twin cases, a robust infrastructure is required to store and process data in real time. Although operating the MQTT broker in a cluster mode can help prevent communication loss from dropped messages, latency may persist due to application processing time problems, especially when dealing with large amounts of data, which can become more complex when using digital twins. A robust infrastructure capable of efficiently storing and processing large amounts of data in real-time according to their specifications is needed to address this challenge. Therefore, a system with the capacity for storing and processing vast amounts of data at runtime is required. The proposed architecture introduces Apache Kafka, which can handle a high volume of real-time data inputs and be distributed, partitioned, and replicated (D'SILVA *et al.*, 2017), which allows for reliable computing on a larger scale of various types and data loads, enabling the use of real-time applications for large data volumes (SOUSA *et al.*, 2018).

The Open Digital Twin organization on GitHub hosts all repositories related to the implemented architecture[3]. It is freely accessible to anyone with GitHub access. The software architecture is designed as a collection of individual microservices and is available under the GNU General Public License v3.0 as an open-source solution. Fig. 4.3 illus-

[3]GitHub organization containing all repositories related to the project found at `https://github.com/Open-Digital-Twin`

trates the proposed architecture. EMQX was selected as the MQTT broker for this project due to its ability to handle high throughput rates (KOZIOLEK; GRÜNER; RÜCKERT, 2020). The system will consist of a cluster of MQTT brokers running in cluster mode, with an analysis performed on the system while (1) receiving messages from multiple IoT devices. Apache Kafka will receive the data through (2) an MQTT Source connector connected to the cluster of brokers for processing (3). The results will then be returned to the MQTT cluster via an MQTT Sink connector (4), serving as data output from Kafka back to the MQTT cluster of brokers, which returns the preprocessed data to the twin instance (5).

Figure 4.3 – Simplified proposed system architecture.



Source: The Author

The upcoming work objective is to assess communication performance between IoT devices and the cloud, specifically through MQTT communication for message transmission and Apache Kafka for stream processing, implementing the system in an edge computing context, utilizing clustering brokers. Furthermore, the study will include an analysis of the features of these tools, such as clustering and data partitioning, focusing

on testing the integration of MQTT and Apache Kafka for IoT communication and processing. Overall, the study aims to provide insights into the effectiveness of this approach for handling large volumes of IoT data in real-time, thereby facilitating the development of efficient and reliable IoT systems.

4.3.1 Clients and Digital Twin Instance

For experimentation, a simulation of various types of devices was created, using client implementations that send pre-defined messages, spawning additional threads for multi-communication via several connections to the MQTT brokers, and sending several messages with configurable frequency and payload. The clients simulate data acquisition to the MQTT broker, representing the "real" aspect of the system as data sources.

The digital twin instance communicates with the message broker, stores data, and defines actions for the system based on the inputs. It is the "digital" part of the twin system. It is a prototype that handles data and connects the physical product throughout its lifecycle (GRIEVES; VICKERS, 2017).

4.3.2 MQTT Broker

EMQX version 5 was chosen as the MQTT broker in the system because it is a high-performance and open-source option, sustaining millions of concurrent connections with low latency and high availability (EMQ INC., 2023). It supports and is easy to implement in a cluster environment, with multiple instances working together in a distributed environment. EMQX can be deployed in various environments, including on-premise, cloud, and edge devices. It also supports integration with other systems and platforms, such as Apache Kafka, Apache Spark, and InfluxDB, through connectors and plugins. In the architecture, it serves as the bridge between the data sources and the digital twin instances.

Several communication protocols and architectures can be used for data acquisition in systems like DTs, including Hypertext Transfer Protocol (HTTP), Constrained Application Protocol (CoAP), Advanced Message Queuing Protocol (AMQP), and MQTT. This study selected MQTT due to its lightweight nature, cost-effectiveness, and compatibility with low computational power devices such as sensors, particularly in environments

where devices are intermittently connected to the network (BANKS *et al.*, 2019). Other protocols like CoAP, HTTP, and AMQP are viable options. However, MQTT was chosen due to its lower latency, minimal network demands, superior security, and widespread use in existing IoT systems (HUMAN; BASSON; KRUGER, 2021).

One solution to address the communication loss resulting from dropped messages is to operate the MQTT broker in a cluster mode, whereby multiple brokers are connected and work together. This approach offers several advantages, including enhanced performance through increased processing power, greater availability due to the redundancy and failure resistance of the system, and efficient load balancing through the distribution of processing tasks across the cluster.

4.3.3 Apache Kafka

The proposed architecture addresses the challenges mentioned above by incorporating Apache Kafka, a high-performance distributed streaming platform, to handle large volumes of data in real time. Kafka's ability to distribute, partition, and replicate data allows for reliable communication of various data types and loads, enabling computing at a larger scale. Kafka's efficient handling of high data inputs in real-time also makes it possible to use real-time applications for processing large volumes of data (D'SILVA *et al.*, 2017; SOUSA *et al.*, 2018). Kafka has a concept of topics that can be partitioned and replicated, ensuring fault-tolerant storage for incoming data streams.

- Broker: Handles all client requests (produce, consume, and metadata) and keeps data replicated within the cluster. There can be one or more brokers in a cluster.
- Zookeeper: Keeps the state of the cluster (brokers, topics, users).
- Producer: Sends records to a broker.
- Consumer: Consumes batches of records from the broker.

A Kafka cluster consists of one or more servers (Kafka brokers) running Kafka. Producers are processes that push records into Kafka topics within the broker. A consumer pulls records off a Kafka topic. Running a single Kafka broker is possible, but it only gives some of the benefits that Kafka in a cluster can provide, such as data replication.

4.3.4 Connectors

Kafka Connect is a powerful tool designed to stream data reliably and efficiently between Apache Kafka and other data systems. With Kafka Connect, defining connectors that move large data sets in and out of Kafka is straightforward. This tool enables the ingestion of entire databases or collection of metrics from application servers into Kafka topics, making the data readily available for stream processing with low latency. Additionally, Kafka Connect includes an export connector capable of delivering data from Kafka topics into secondary indexes such as Elasticsearch or batch systems such as Hadoop for offline analysis. Overall, Kafka Connect provides a highly scalable and efficient solution for streaming data between various data systems, making it a valuable tool for building and managing data pipelines.

In this case, Kafka Connect connects the MQTT brokers to and from Kafka through source (to inject data into Kafka from MQTT) and sink (to inject data from Kafka back to MQTT) connectors. Kafka Connect MQTT Source Connector connects to an MQTT broker and subscribes to the specified topics. Any received data is then written back into Kafka topics. On the other hand, Kafka Connect MQTT Sink Connector subscribes to a specified Kafka topic, writing its data to an MQTT broker. A JSON (JavaScript Object Notation) is required to define the connectors between services, allowing for quick and easy deployment of custom connectors for numerous services.

4.3.5 Processing Kafka Data

A data processing system like Kafka stores and aggregates events coming from several different sources in real time, and it is used to perform specific computations on time. It is excellent for creating scalable ingestion of event streams from many producers to many consumers. Kafka itself is just a distributed messaging system, but it can be used to enable other data processing systems to process its data, including:

- Kafka Streams: is a Java library for building real-time, distributed, and fault-tolerant data processing applications that process data in Kafka topics. It uses the Streams API (Application Programming Interface) from Kafka to receive data, processing the topics via streaming, a continuous flow of data records incoming and processing them as they arrive, internally using producer and consumer libraries

(BEJECK, 2018).

- Apache Spark: is a distributed memory-based data transformation engine. It can connect with several databases and file systems while operating in distributed environments, enabling parallel processing between devices. One of the many extensions to the core Apache Spark ecosystem is Spark Streaming, which uses a micro-batch processing model, processing small batches of data records collected over time and periodically checking the source for a batch of data to process, which may cause higher latency than via streams since records must wait until the subsequent batch request is processed. Spark also offers an alternative processing model called Continuous Streaming, which does not collect batches and sends data to be processed as received (HALL, 2018; RATHI, 2023).

Both are prevalent solutions that can be used to process Kafka data. In short, Kafka Streams offers a more straightforward API to connect and process Kafka topics, while Spark requires more planning but can support multiple other data source types. This decision is dependent on the data processing requirements. Other popular choices in the Apache software ecosystem are Storm and Flink, each with its focus. In the case of this implementation, a custom handler application was used, implemented using node-kafka, a high-performance NodeJS client for Apache Kafka that wraps the native *librdkafka* library (BLIZZARD, 2023), to create a custom consumer and producer container. The *librdkafka* The handler application supports communication via several different methods provided by Kafka, using both the Stream API, which receives a stream of data and processes it as it gets received, and the Standard API, with two types: the *flowing* method, which maintains an infinite event loop in the program and receiving messages until the connection is disconnected, and the *non-flowing* method, which gradually reads messages, by request.

5 EXPERIMENTS

Based on the system architecture presented in Chapter 4, several experiments were made to provide insight into the potential usage of an MQTT and Kafka-based Digital Twin architecture.

5.1 Cluster Configuration

The experiments were executed using a Kubernetes cluster hosted in the internal network of the Institute of Informatics, formed by several different types of machines, including several Raspberry Pi 3 and 4 to be used as edge devices, regular desktop computers, and higher capacity server-grade machines. Previous experiments made for other research papers were made using simulations with virtual machines, which, even though they provide valuable data, lack in representing deployments in real-life scenarios due to only having a simulated network, where data loss in transmission is non-existent, and all communication is perfect.

In this cluster, a real network is created between different physical computing devices, where data is transmitted and passes through real network interfaces, with physical cables transmitting bits between switches, more realistically emulating all the added delays and protocol handling a network would face in a real-life scenario. Previous research using this architecture executed experiments in a simulated environment, with virtual machines emulating different devices (KNEBEL; WICKBOLDT; FREITAS, 2020; KNEBEL, 2020; TREVISAN; KNEBEL; WICKBOLDT, 2020; TREVISAN; KNEBEL; WICKBOLDT; ABEL, 2022). Although that scenario offers many possibilities and allowed for initial research, the main idea was performing future experiments in a deployment that more accurately resembles what real industries would have to face when trying to implement a system-of-systems like Digital Twins in a production environment: different devices, with differing hardware between them and all the problems that a multilayer microservice-based faces in a current edge/cloud computing implementation.

For the experiments that are going to be presented (and others done by the study group), a Kubernetes cluster was created, which as of *September 9th, 2023*, as shown in Fig. 5.1, aggregates a total of 432 CPUs, 696.7 GiB of RAM and 7.0 TiB of disk space, distributed between 46 different nodes in the cluster. The nodes are labeled with three different classifications: “computer”, “edge” and “server”. Those are used in experiments

Figure 5.1 – Custom script that extracts the total capacity from the cluster nodes.

```
fpknebel@blacksabbath:~/test-kafka-emqx-k8s$ python3 cluster-capacity.py
computer {'cpu': 72, 'memory': 123888752, 'disk': 5095140440, 'nodes': 16}
edge {'cpu': 80, 'memory': 39764208, 'disk': 839742960, 'nodes': 20}
server {'cpu': 280, 'memory': 566886436, 'disk': 1614571128, 'nodes': 10}

Total CPUs: 432
        computer: 72
        edge: 80
        server: 280
Total Memory: 696.7GiB
        computer: 118.1GiB
        edge: 37.9GiB
        server: 540.6GiB
Total Disk: 7.0TiB
        computer: 4.7TiB
        edge: 800.8GiB
        server: 1.5TiB
Total Nodes: 46
        computer: 16
        edge: 20
        server: 10
```

Source: The Author

to limit pods to run in specific machines that have hardware matching the real-life counterpart device we simulate in the experiments. For example, the “edge” label includes three Raspberry Pi 4 and seventeen Raspberry Pi 3 devices, which have lower hardware specifications compared to the “computer” and “server” machines.

5.1.1 Containers

Containers are used for packaging software and its dependencies into a portable unit running consistently across different computing environments. An application inside a container includes all its libraries, settings, and tools in a self-contained and isolated environment. It is somewhat of a “specialized virtual machine” where the developer does not have to worry about compatibility issues with other hardware or software and focuses only on the application itself. This portability eliminates the need to adjust applications to different environments and ensures consistency and reliability.

The need for containers can be explained as purely economical. Considering a pool of infinite computing resources, one can build any computing system to support an equally infinite demand, only depending on its application logic (and the talent applied by its programmers). Unfortunately for us who abide by the rules of physics, infinitely scaling an application is arduous. Considering the strict usage of physical servers, the

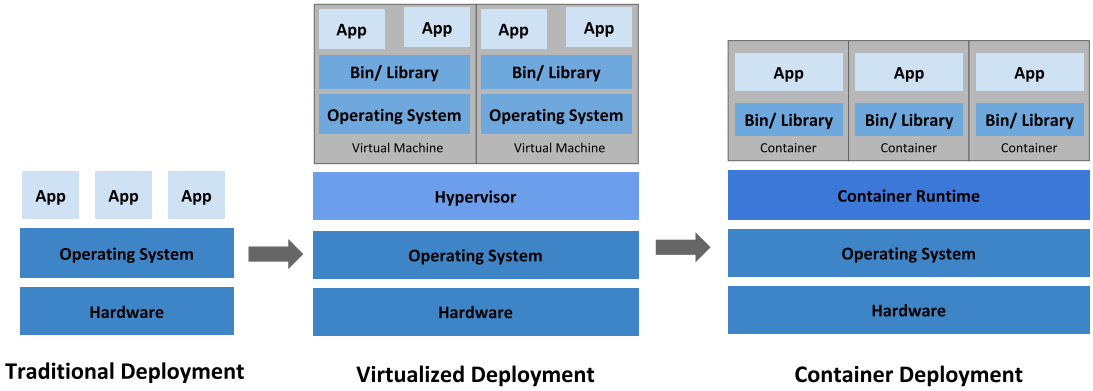
allocation of resources would quickly depend on the limits defined by the resources available on-premises, and building new servers to handle a temporary surge is only possible without even attesting to its implementation viability. Organizations needed a means to define rules for allocating resources between multiple applications, leading to instances where one application would consume a disproportionate share of resources, resulting in decreased performance by the others. Running each application in separate physical servers is not recommended, as it could be more scalable and cost-effective since resource sharing would still be unmanaged, leading to unused resources and high maintenance costs.

Virtualization then shifts as a possible alternative: instead of having to install new machines for every project, now we can just let the software handle it, creating virtualized servers inside the existing one, sharing resources, and allowing for several applications to run simultaneously. Virtualization also solved the isolation problem: virtualized applications ran on another layer, unaware of the other applications running inside the server and preventing unrestricted access to data of one application by another. Virtualization enhances resource utilization within a physical server, improving scalability by facilitating easy addition or update of applications. Each virtual machine is a fully independent machine that functions just like a physical one, including its operating system, on top of the virtualized hardware. By allocating a set of physical resources as a cluster of disposable virtual machines, virtualization enables the creation of a more flexible and dynamic computing environment. Unfortunately, virtualization depends on the hypervisor abstraction layer to manage its virtual machines, requiring additional CPU, memory, and disk space, which implies generating an additional resource overhead. Virtualized environments can also become very complex to maintain, especially as the number of virtual machines grows, which may lead to configuration errors and security vulnerabilities.

Overall, while virtualization has been a valuable technology for creating isolated environments, its drawbacks have led to the increasing adoption of containerization. This more lightweight approach still offers many advantages. Fig 5.2 illustrates the differences between traditional, virtualized, and containerized approaches.

Containers are a form of virtualization that enables applications to run in isolated environments on top of a shared host operating system. They package an application and its dependencies, including libraries, into a single executable unit that can run consistently across different computing environments. Containers allow developers to create portable applications that can be easily moved between different systems without worrying about

Figure 5.2 – Traditional, Virtualized, and Container deployment differences.



Source: The Kubernetes Authors (KUBERNETES, 2023)

dependencies or configuration issues. Compared to traditional virtual machines, containers are lightweight and consume fewer resources, making them ideal for modern application deployment scenarios such as microservices architectures and cloud-native applications, as they are decoupled from the underlying infrastructure, making them portable across clouds and OS distributions.

A container image is a software package with everything to run an application: the code, the runtime, the libraries, and the settings. Containers are designed to be stateless and immutable, meaning that any warranted changes to the application require building a new image, inserting those changes into a new image version, and requiring the container to restart using the new image.

5.1.2 Kubernetes

Knowing the context and what containers are used for, Kubernetes is a platform that orchestrates multiple containers. With the rising complexity of containerized applications, Kubernetes manages these services, creates and scales containers automatically, and organizes required resource allocation, like storage. It can identify increased traffic in the application and scale containers up or down, efficiently restarting, replacing, or terminating containers that fail to respond to health checks (GUTHRIE, 2022).

Kubernetes has become a top contender for container orchestration in recent years. While Kubernetes was initially focused on cloud environments, its versatility has made it an attractive option for new use cases that demand reliable, high-performing orchestration in edge computing (JEFFERY; HOWARD; MORTIER, 2021). The popularity of

Kubernetes is on the rise, and it is being increasingly embraced by various professionals in the technology industry, such as DevOps engineers, backend developers, and other tech specialists (ASLAMOVA, 2021).

According to the Cloud Native Computing Foundation Annual Survey 2022, Kubernetes has established itself as a powerful and widely adopted platform for organizations, with its versatility and ability to support various workloads solidifying its position as the defacto "operating system of the cloud." A significant number of respondents, 44%, already use containers for most applications and business segments, and 35% use them for some production applications, with 9% still evaluating implementation (LINUX FOUNDATION, 2023).

For smaller applications, Kubernetes may not be the most suitable option. However, it can be a powerful and flexible choice for larger organizations, fast-growing startups, or companies upgrading legacy applications that have very complex functionalities and need to scale to support many users most efficiently, especially considering the costs of cloud computing services.

5.2 Experiment Configuration

For the experimentation scenarios, a simulation of various types of devices (e.g., sensors) was created using client implementations that send messages in a controlled manner. Each client created a configurable number of threads connected to the MQTT brokers and sent messages over time. The number of threads increases until the client reaches a maximum message rate, then decreases until the program ends. Each client represents a service class for different Digital Twin types, based on Table 5.1, adjusting the data transfer windows and message sizes to satisfy the latency requirements. This table and the classes were inspired by 3GPP's QoS Class Identifier (QCI) (3GPP, 2021) and Maaloul et al.'s work (MAALOUL *et al.*, 2021), and originally defined for (TREVISAN; KNEBEL; WICKBOLDT; ABEL, 2022).

Table 5.1 describes five service classes for different applications: critical mission (high priority), industrial automation with two packet sizes, vehicular (V2X), and non-critical vehicular (V2X Non-Critical). The data window (*Window (ms)*) sets the time between messages for each service. The *Message (bytes)* column shows the packet size for each client type. The *Max Latency (ms)* column shows the maximum allowed delay between sending and receiving a message for each service.

Service	Window (ms)	Message (bytes)	Max Latency (ms)
Mission Critical Application	500	64	75
Discrete Automation (small)	2000	255	10
Discrete Automation (big)	2000	1354	10
V2X	100	64	50
V2X Non-Critical	1000	1200	50

Table 5.1 – Digital Twin Service class characteristics.

Each client spawns a container for each class in the Kubernetes cluster containing a custom-made producer application that generates data representative of the requirements of that class and transmits it to the MQTT brokers. The program was set up in the experiments to create data for an entire hour, scaling a new application thread every 90 seconds until we had 15 threads of that client scaled. To limit the experimentation scope, after a thread exists for an hour, it stops producing new data for that class.

5.3 Experimentation with the cluster

The following subsections define the deployment scenarios employed with the built architecture and discuss their results. Hundreds of iterations of experiments were executed using this architecture, so the following sections will be an aggregation to display only the main results found. Metadata from the experimentation results can be found in the project's GitHub repository [1]. The experiments will present resource usage metrics from the pods through graphs to better understand their performance when stressed in scenarios matching the defined digital twin service class characteristics from Table 5.1.

5.3.1 Experiment 1

A base test case for testing the architecture, using five clients for each service class, with its correspondent twin instance, sends messages at an interval of one hour, with a single MQTT broker and a single Kafka broker instanced. Each client pod generates messages with the class specification received by the MQTT broker. The Kafka connectors then pull from the source topics, saving the messages in a Kafka topic. After processing, data is saved into sink topics, which the sink connectors push back to MQTT

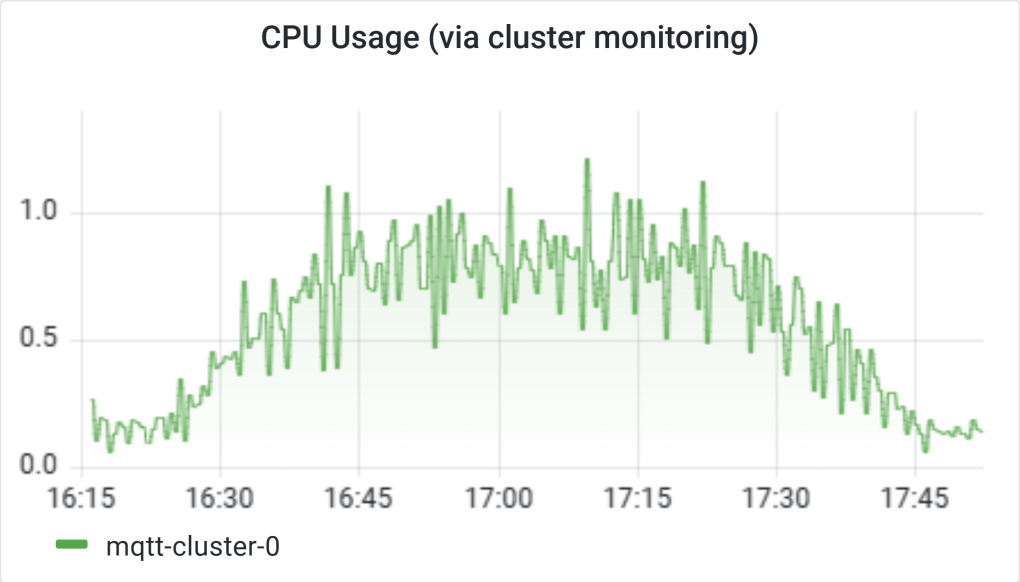
[1] Repository is hosted in the Open Digital Twin organization in GitHub and can be directly accessed at the `https://github.com/Open-Digital-Twin/test-kafka-emqx-k8s`.

so the twin instances can receive their processed data. Experimenting with this basic setup presents several problems up to discussion:

- No tolerance in case of broker failure.
- In the case of the MQTT broker, all queued messages would be lost.
- EMQX can mitigate this with the use of retained messages, which store the **most recent** message sent to a topic [2].

As evident in Fig. 5.3 and Fig. 5.4, the EMQX broker is very stable regarding memory usage while CPU-intensive due to the high amount of processed messages. Fig. 5.5 shows a graph of the total messages sent, received, and dropped by the broker, while Fig. 5.6 presents the broker message rate. In the scenario tested, only one broker is being used to process the messages received, which bottlenecks the rate to around 2000 messages per second. These messages include a) new data being generated by the client application being sent to the broker, b) data being picked from the source topic in the broker to be pushed to Kafka, c) processed data being pushed from Kafka to the sink topic and d) processed data from the sink topic being picked and sent to the twin instances. In this scenario, scaling the number of clients reached a maximum message rate per broker of around 3000 messages per second, as shown in Fig. 5.7.

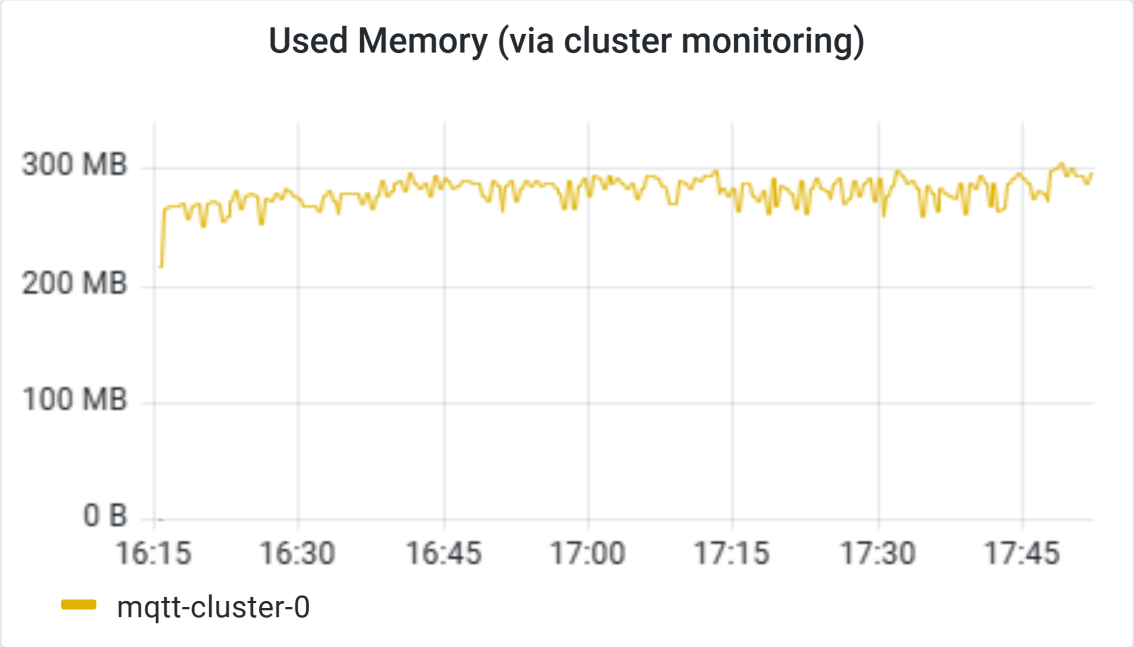
Figure 5.3 – Experiment 1: EMQX broker CPU Usage



Source: The Author

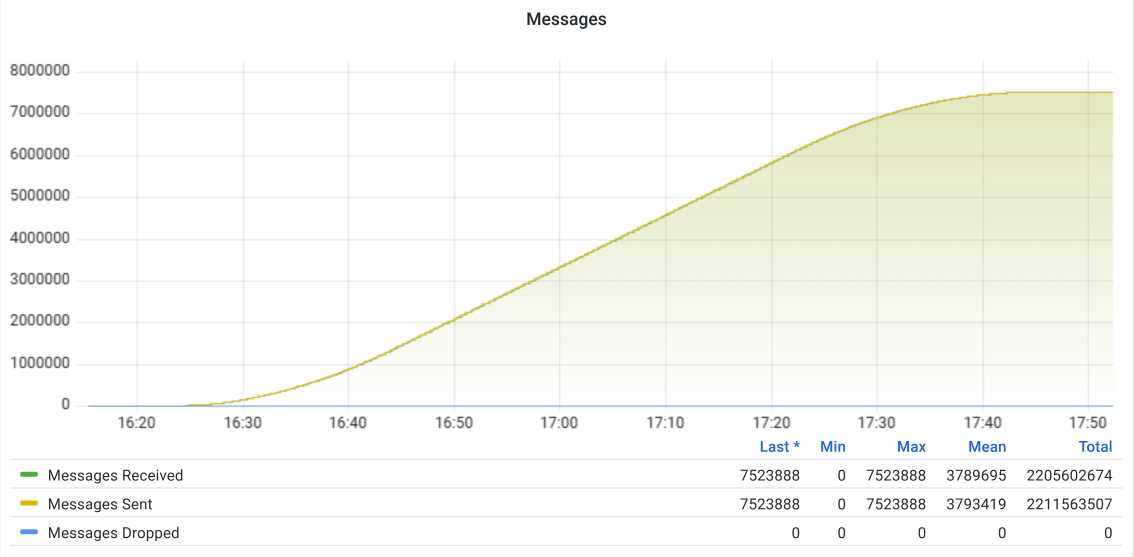
[2]EMQX documentation on MQTT Retained Message: `https://www.emqx.io/docs/en/latest/messaging/mqtt-retained-message.html`

Figure 5.4 – Experiment 1: EMQX broker Memory Usage



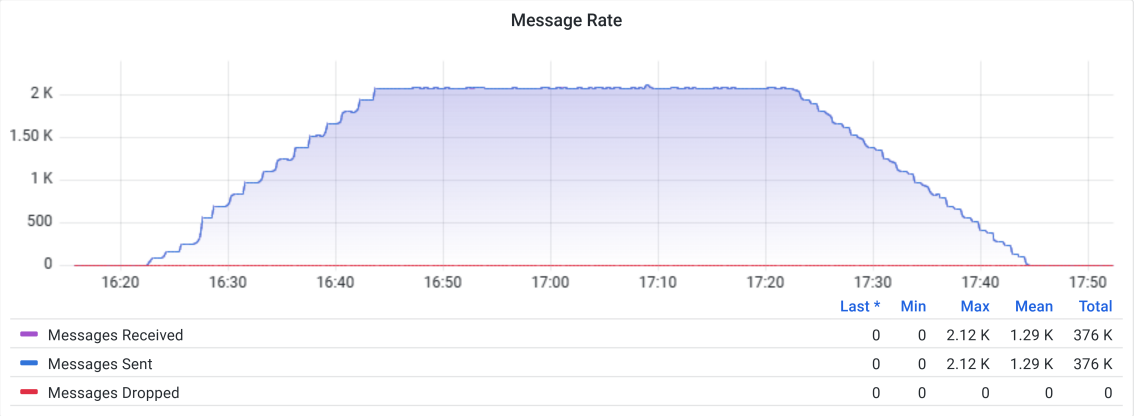
Source: The Author

Figure 5.5 – Experiment 1: EMQX broker message count



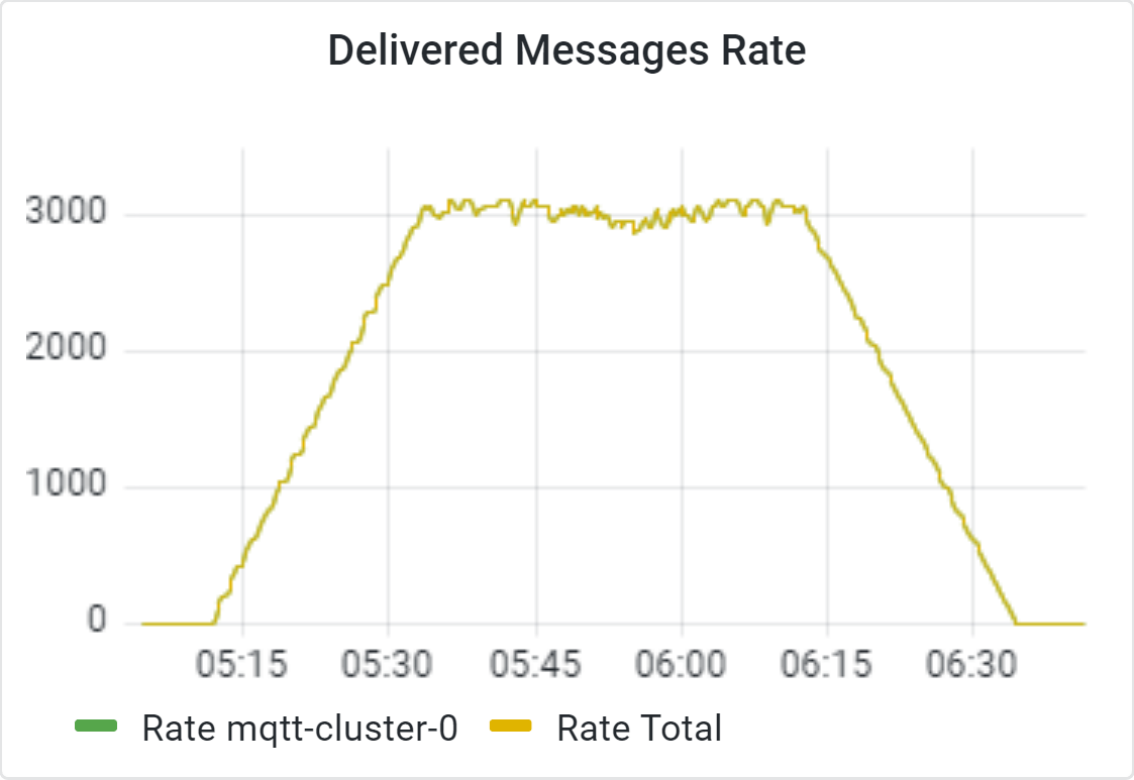
Source: The Author

Figure 5.6 – Experiment 1: EMQX broker message rate



Source: The Author

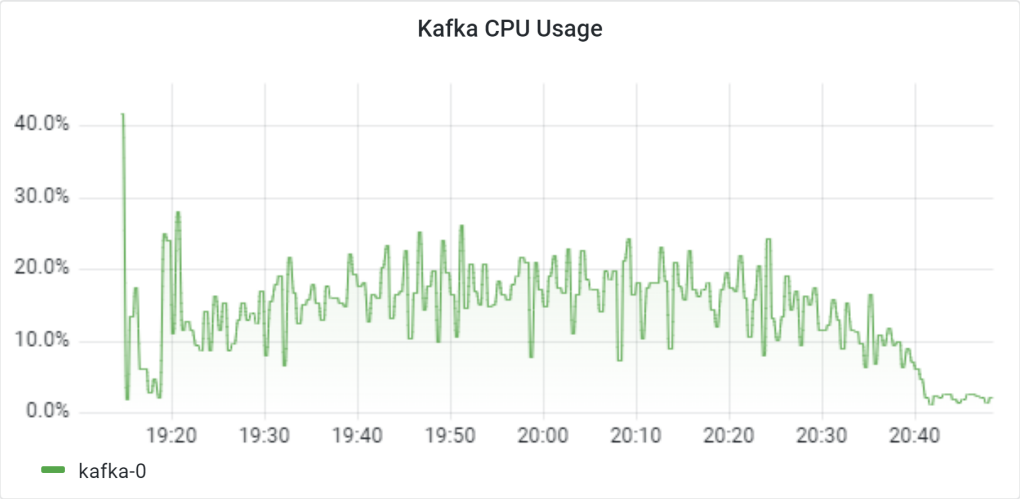
Figure 5.7 – Experiment 1: EMQX broker maximum message rate obtained



Source: The Author

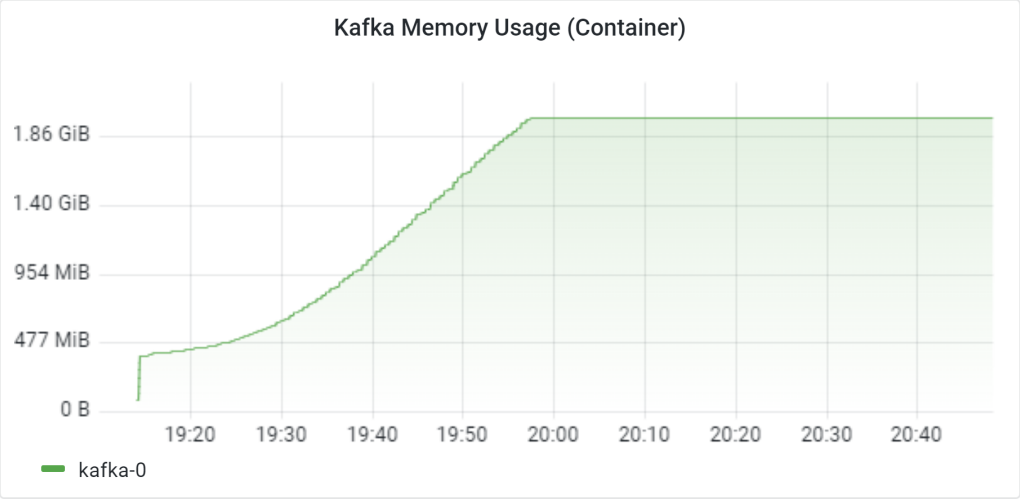
Comparatively, Fig. 5.8 and Fig. 5.9 show the CPU and memory usage of the Kafka instance, where a very different behavior is seen due to caching of messages. The MQTT cluster does not need to store most messages for long: they are received, parsed, and forwarded to clients subscribed to that topic, clearing the memory of that message. Kafka, on the other hand, wants to cache the entire log of messages in memory for faster retrieval, backtracking and accessing past messages, which will scale to using as much memory as it can before clearing messages from memory, which is visible due to the pods being force-limited to use at most two GBs of memory.

Figure 5.8 – Experiment 1: Kafka CPU usage



Source: The Author

Figure 5.9 – Experiment 1: Kafka memory usage

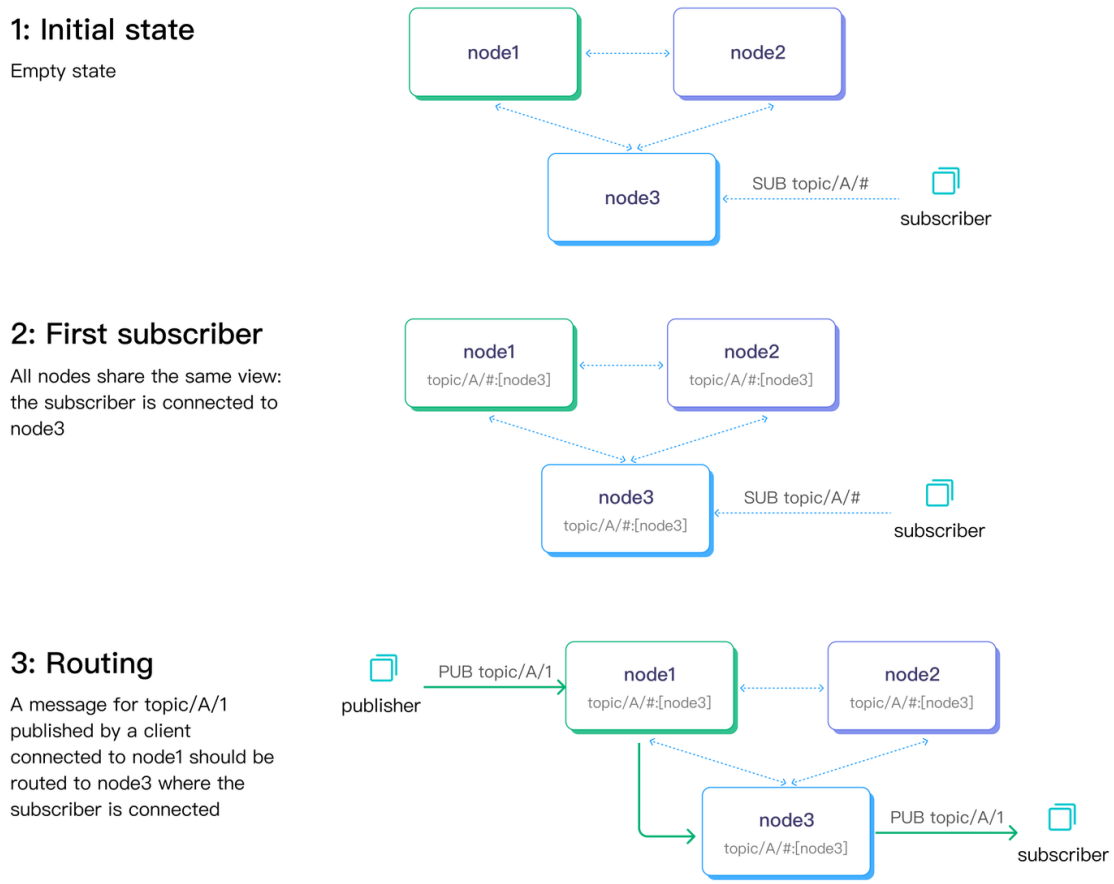


Source: The Author

5.3.2 Experiment 2

EMQX supports clustering the broker into multiple connected nodes. In an EMQX cluster, there are two data replication channels: metadata, which shares routing information of what topics are being subscribed to by which nodes, and message delivery, which forwards messages between nodes. Fig 5.10 illustrates how messages are forwarded through the cluster nodes from publisher to subscriber.

Figure 5.10 – EMQX clustering data replication channels.

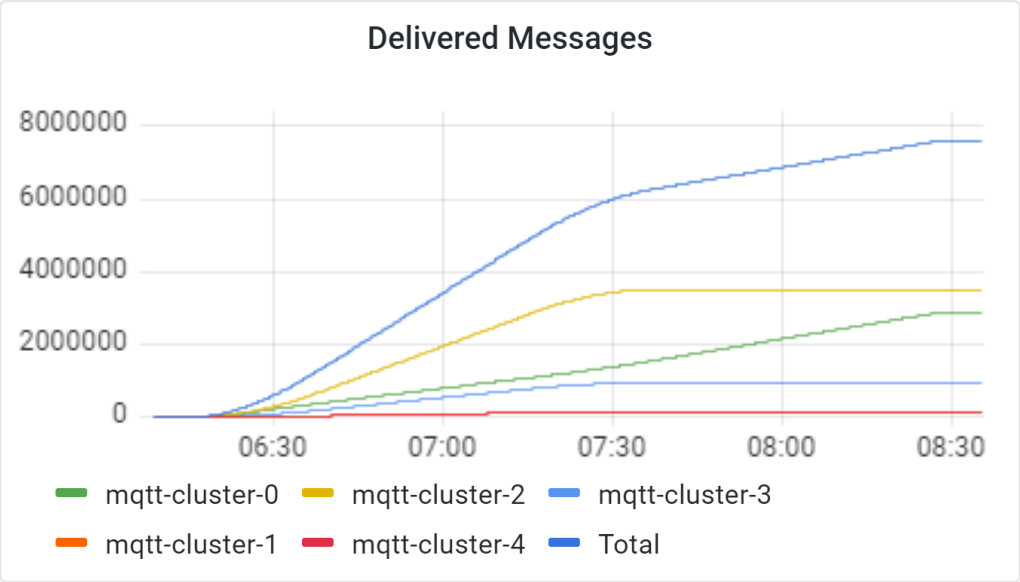


Source: (EMQ INC., 2024)

One run of the experiments performed while clustering the MQTT broker used 5 EMQX brokers connected in this default method provided by EMQX. Comparing the message rates from Fig. 5.11 and Fig. 5.12 to the previous experiment Fig. 5.6, considering both experiments have the same amount of clients generating data at the same rates, the random nature of the distribution of topics in the cluster generates a situation. Broker ‘mqtt-cluster-0‘ adds a long tail to the experiment runtime, adding an extra hour until all messages are sent to the subscribers.

Investigating the reason for this scenario is explained by the packet count separated by each broker, presented in Fig. 5.13. Considering how a subscriber is connected to one exact broker, the generated messages were being sent to ‘mqtt-cluster-3‘ in this case. Knowing how the data replication channels of the EMQX broker work, all broker nodes know the subscribed topics in all the other brokers, so after receiving the messages, the broker pushes these messages to ‘mqtt-cluster-0‘, which then transmits the message to the intended subscriber.

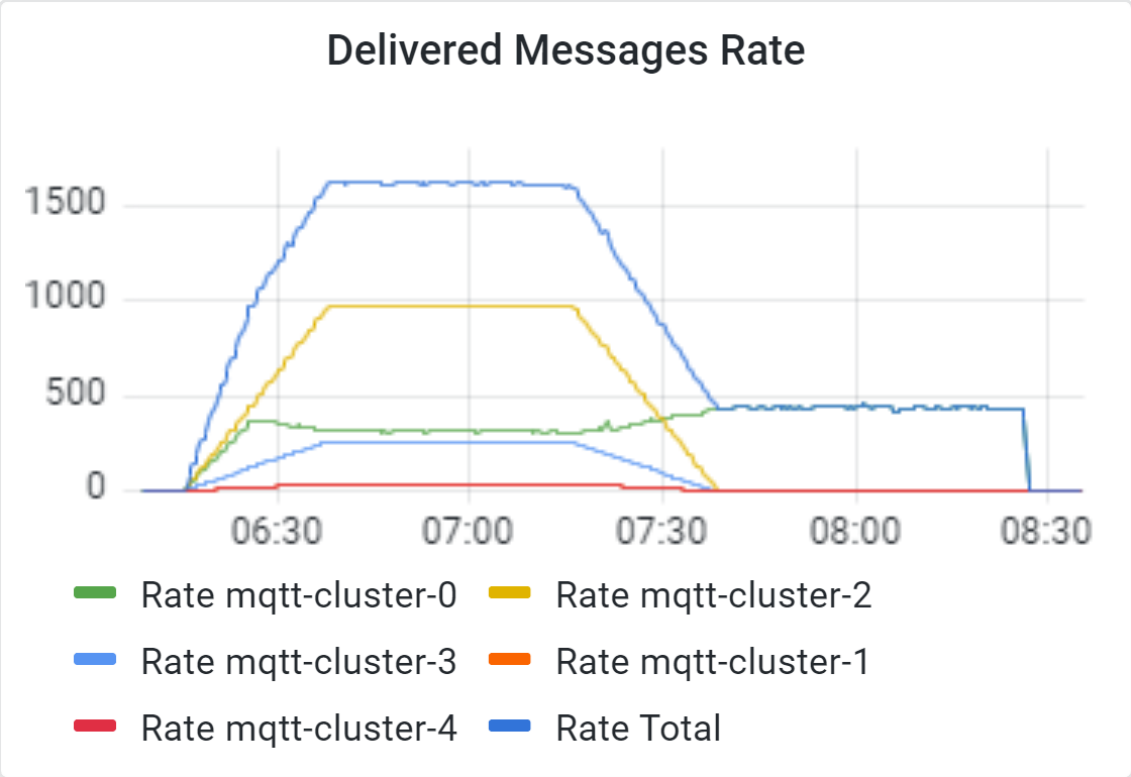
Figure 5.11 – Experiment 2: EMQX Delivered Messages



Source: The Author

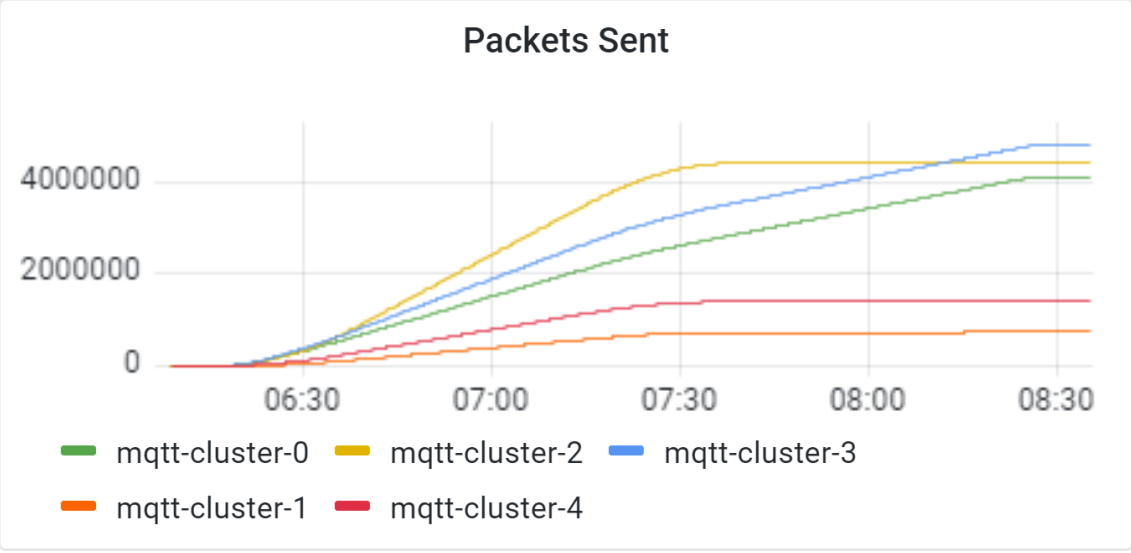
This form of execution of the broker cluster presents a limitation in the communication since the broker is responsible for both the handling of its clients and the retransmission to other brokers, degrading the application’s performance. Unfortunately, this situation is not easily solvable without previous knowledge of the entire scope of connections or with a protocol change for the broker to solve this issue independently. If the protocol allowed for it, what could be done in the background was to force a reconnection to the direct broker, either between the client publishing the data or the client subscribing to it, so that retransmission across brokers was unnecessary. This generates another difficult problem: how do you handle clients who are subscribed to the same topic but are connected to different clients? EMQX allows this, transmitting the message to both brokers, but this will generate additional traffic multiplied by the distribution of subscribers per broker in that topic.

Figure 5.12 – Experiment 2: EMQX Message Rate



Source: The Author

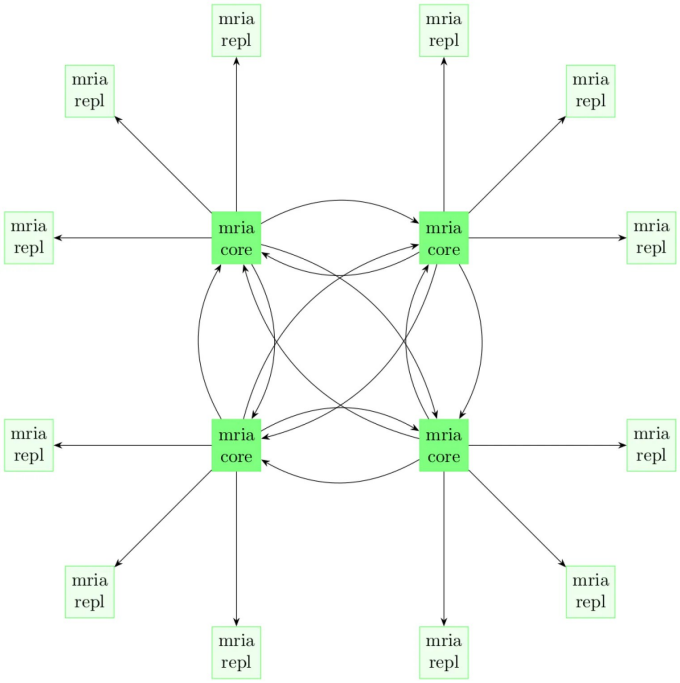
Figure 5.13 – Experiment 2: EMQX Packets Sent



Source: The Author

EMQX in versions before version 5 only had the option of using a full mesh topology, storing all topic subscriptions in all broker nodes, which scales the number of connections at a factor of N², as shown in Fig 5.10. Because of this, as the quantity of nodes and replicas increases, so does the overhead for coordinating write transactions and the risk of encountering split-brain scenarios (EMQ INC., 2022). Post version 5, an extension to the cluster implementation was introduced, creating two node types called Core and Replicant (WARE, 2022):

Figure 5.14 – EMQX cluster Core and Replicant node topology example



Source: `https://docs.emqx.com/en/emqx-operator/2.1.0/tasks/configure-emqx-core-replicant.html`

- **Core** nodes: which act the same way as the previous implementation.
  - Connect to other Core nodes in full mesh;
  - Table data synchronized across nodes;
  - Same scaling node problem, but with a reduced set of nodes.
- **Replicant** nodes: only one connection, connected to one Core node.
  - Connects to a Core node only;
  - Acts like a client to a Core node;
  - Data is updated asynchronously;
  - Scales by N instead of N².

5.3.3 Experiment 3

A problem found when dealing with multiple Kafka brokers was the need for extra configuration to work with ZooKeeper, which is different for each broker. This required changing how the deployment rules were made to connect the Kafka broker correctly in the cluster. This motivated using a new feature from Kafka: the Apache Kafka Raft (KRaft), a new self-managed metadata quorum controller protocol. KRaft works as follows:

- KRaft ensures that the quorum controllers replicate metadata correctly across the cluster.
- The state is stored in an event log (the metadata topic), regularly shortened by snapshots to prevent the log from growing too large.
- Non-leader controller in the quorum follow the active controller by reacting to the events it produces and saving them in its log. Therefore, if one node loses connection, it can catch up on any missed events by reading the log when it reconnects.
- When leadership changes, the elected controller already has the full state in memory, reducing downtime.

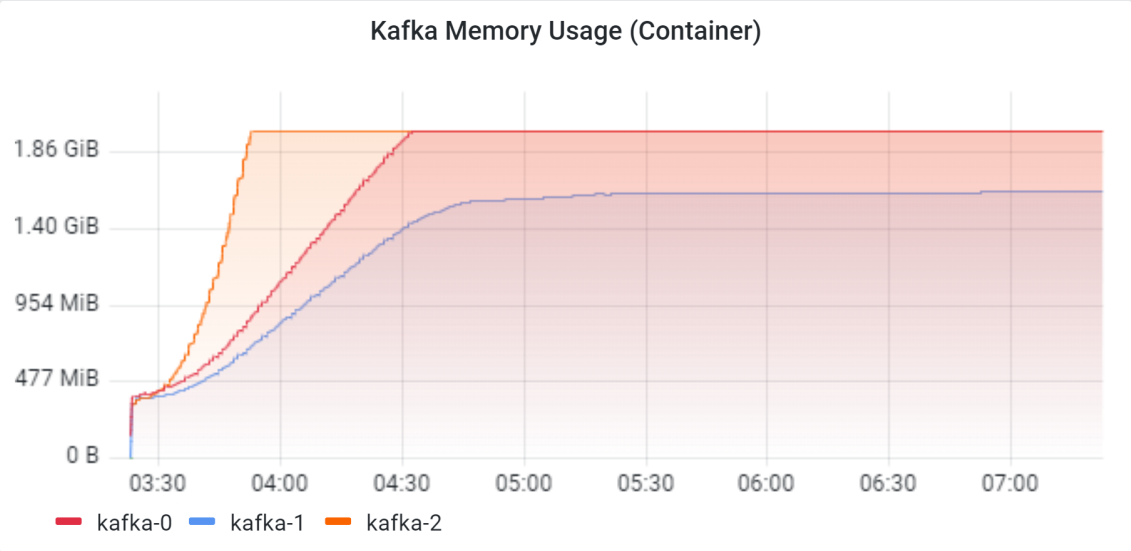
ZooKeeper is not built to accommodate a high volume of clients or requests. As the size and traffic of Kafka clusters increase, ZooKeeper becomes the bottleneck, impacting Kafka's performance and availability while introducing an additional layer of complexity and dependency to Kafka's structure. Those using and managing Kafka must install, configure, monitor, and troubleshoot ZooKeeper separately. Furthermore, ZooKeeper has its own configuration parameters, failure scenarios, and security protocols that differ from Kafka's. ZooKeeper ensures strong consistency, meaning all nodes in the cluster have the same data view at any given moment. However, this also implies that ZooKeeper needs a quorum (majority) of nodes available to process any request. If a quorum is unavailable, ZooKeeper cannot serve any request and becomes unavailable. This can, in turn, affect Kafka's availability, particularly in situations like network partitions or data center failures (GLUSHACH, 2022).

This new protocol simplifies Kafka's architecture by consolidating responsibility for metadata into Kafka itself. It improves stability, simplifies the software, and makes monitoring, administering, and supporting Kafka easier while removing an external dependency (CHANDRAKANT, 2024; MCCABE, 2020). Previously, Kafka used

ZooKeeper to store its metadata about partitions and brokers and elect a broker as the cluster controller. The idea is to remove this dependency and enable metadata management more scalable and robustly by consolidating responsibility into Kafka while simplifying its deployment and configuration. After the Kafka 4.0 release (expected for April 2024), only the KRaft mode will be supported, with ZooKeeper being deprecated since Kafka 3.4 (January 2023). These changes make Kafka more powerful, improve its stability, simplify its software, and make monitoring, administering, and supporting more accessible.

Extending from the problems of the previous experiment, the third experiment setup was as follows: Three clients for each service class, with its corresponding twin instance, three MQTT clustered brokers, and now using three Kafka instances running on KRaft mode. In this multi-broker approach, the memory distribution between brokers is more stable, as seen in Fig. 5.15. In Fig. 5.16, you can see that the “kafka-2” instance was elected as the controller, explaining the increased CPU usage and reaching the memory limit sooner.

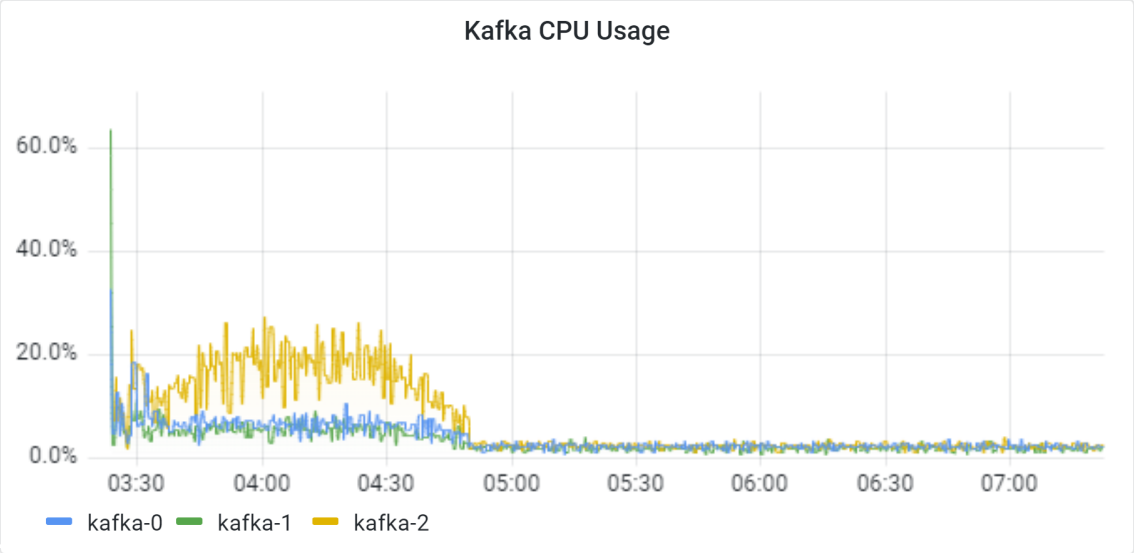
Figure 5.15 – Experiment 3: Kafka Memory Usage



Source: The Author

The problem in this experiment lies in the growing demand for resources. In Fig. 5.17 and Fig. 5.18, the CPU and memory usage for the Kafka Connect instances are presented, responsible for bringing the source topics to Kafka from EMQX and the sink topics from Kafka back to EMQX. These hard limits at 2 GB per instance were set to make execution possible but are not ideal, as they limit the application’s performance. The limits were quickly reached in the first 15 minutes of the experiment, slowing down

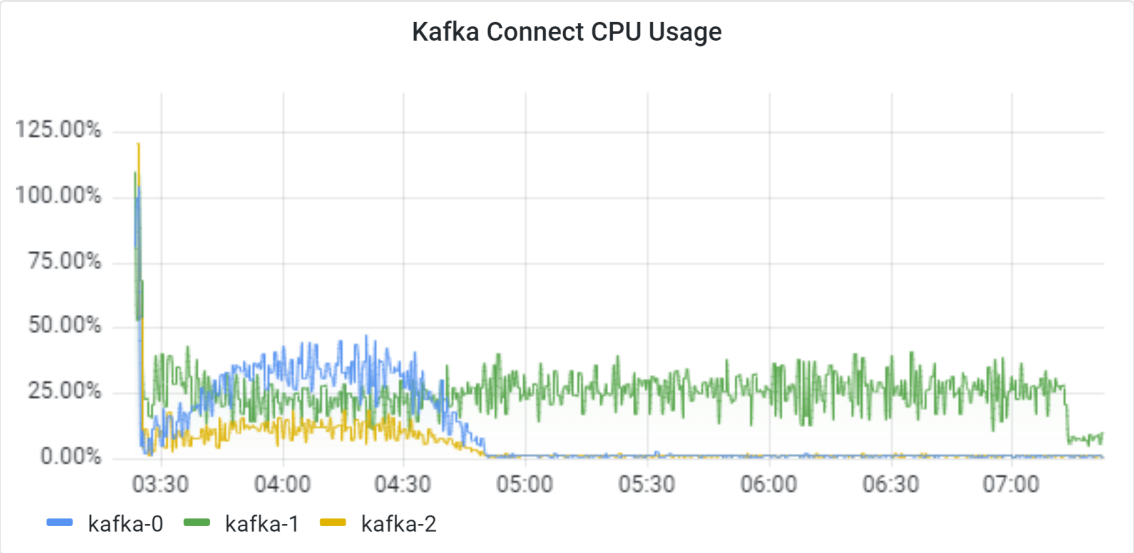
Figure 5.16 – Experiment 3: Kafka CPU Usage



Source: The Author

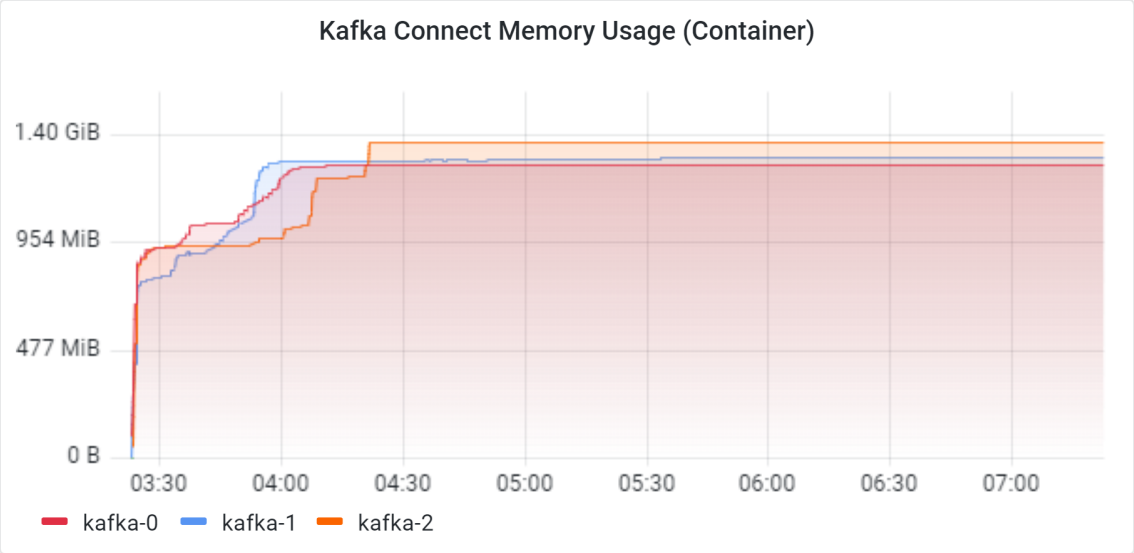
the message rate. In Fig. 5.19, the packets sent between MQTT brokers, as discussed in Experiment 2, show that the long tail problem of communication between brokers is still present, heavily extending the required time until all messages are delivered.

Figure 5.17 – Experiment 3: Kafka Connect CPU Usage



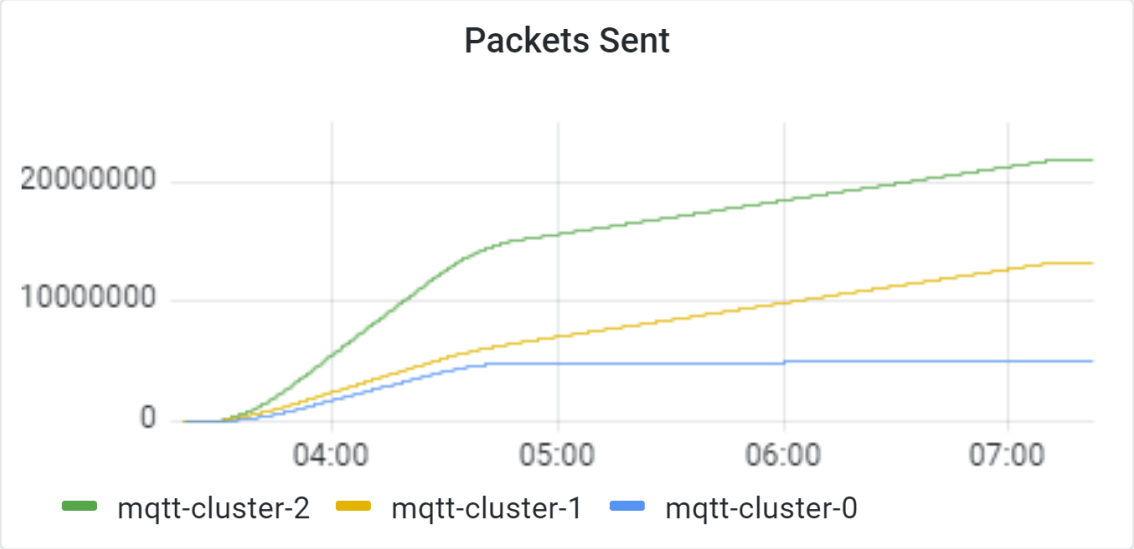
Source: The Author

Figure 5.18 – Experiment 3: Kafka Connect Memory Usage



Source: The Author

Figure 5.19 – Experiment 3: EMQX broker packets sent



Source: The Author

6 KAFKA FOR DIGITAL TWINS

Even though Apache Kafka (or similar platforms) offers several benefits to communication, it is vital to research and validate if adding all these elements to the architecture will benefit the end user. Is it worth guaranteeing data consistency if it represents a very outdated state of the client sending messages when receiving the processed data? Is it better for the system to support some message loss but be able to support more data entries generating more recent data? From the results of Chapter 5, the following chapter will discuss challenges and solutions to the usage of Apache Kafka in the context of Digital Twins.

6.1 Challenges of Kafka usage

- Edge Deployment: Edge computing often involves low-footprint, low-touch, little-or-no-DevOps-required installations. Apache Kafka, on the other hand, is very resource-intensive and requires significant DevOps effort to install and operate efficiently. For instance, many edge computing scenarios would require deploying Kafka brokers across hundreds of locations, often without IT experts on-site to operate Kafka, making managing and maintaining the system more complex. Edge computing often requires offline business continuity, even when the connection to a central data center or cloud is unavailable. This means that the system must be able to operate independently and continue processing data even when disconnected from the main network.
- Setup Difficulty: Setting up and managing Apache Kafka is not that simple: figuring out networking requirements, setting up the proper interfaces, and segregating in terms of security. And then, after having it all up and running, knowing how to diagnose and resolve problems as they arise is needed.
- Learning Curve: Developers new to Apache Kafka might struggle to grasp the concept of Kafka brokers, clusters, partitions, topics, and logs. The learning curve is steep, and extensive training is needed to learn Kafka's basic foundations and the core elements of an event streaming architecture. Understanding these concepts and their interaction can be challenging, especially for developers new to distributed

systems and messaging.

Therefore, while Apache Kafka has many strengths, if the use case cannot provide the required hardware and personnel, it is not a universal solution that is the most suitable choice for all edge computing scenarios due to these challenges.

6.2 How to effectively use Kafka for digital twins

Apache Kafka allows defining topics (a topic is similar to a category name to which records are stored and published), where applications can add, process, and reprocess records. Applications communicate with this system and send a record to a topic. A record can contain any information, for instance, information about an event that occurred on a website or an event that triggers another event. Another application can connect to the system and process or reprocess records from a topic. The data sent is kept until a specific retention period expires.

Records are byte arrays that can store any object in any format. A record has four attributes: key and value are mandatory, and the other attributes, timestamps, and headers are optional. The value can be whatever needs to be sent, for example, JSON or plain text. As mentioned before, all Kafka records are grouped into topics. Producer applications write data to topics, and consumer applications read from topics. Records published to the cluster remain in the cluster until a configurable retention period passes by.

Kafka retains records in the log, making the consumers responsible for tracking the position in the log, known as the “offset”. Typically, a consumer linearly advances the offset as messages are read. However, the consumer controls the position, which can consume messages in any order. For example, when reprocessing records, a consumer can reset to an older offset.

6.2.1 Topic Partitioning

Data consistency in Kafka can be achieved by using several mechanisms. One option is to divide the topic across multiple brokers. Any record written to a particular topic goes to a particular partition. Each record in a partition is assigned and identified by its unique offset. A topic can also have multiple partition logs, which allows multiple

consumers to read from a topic in parallel. The logic that decides the partition for a message is configurable, helping parallel reading/writing of data in different partitions spread over multiple brokers.

In a partition, one broker is chosen as the leader, while the remaining brokers with replicas are designated as followers. Kafka must account for potential issues such as followers lagging behind the leader or the leader becoming unavailable due to crashes or restarts. To manage this, Kafka differentiates between followers who can stay updated with newly appended records and those who cannot. The up-to-date replicas form a subset of all the partition's replicas, referred to as the in-sync replicas (ISR). A follower maintains its in-sync status as long as it consistently fetches the most recent records. When a producer aims to append new records, it should anticipate an acknowledgment from the leader that confirms the successful append operation while being prepared to resend the messages if it fails to acknowledge the transmission success. Suppose a follower partition fails to maintain synchronization (a lag timeout broker configuration can be defined for this). In that case, the leader removes it from the ISR, logging it in the partition metadata. If the leader crashes, only another replica from the ISR can be elected as the new leader. If a follower fails to make sufficient fetch requests to maintain synchronization with the leader, the leader removes it from the ISR. This modification is logged in the partition metadata within the control plane and is governed by the "replica.lag.time.max.ms" broker configuration parameter. If the leader crashes or is shut down, the control plane will appoint another replica from the ISR to take over as the new leader. Kafka allows producers to specify how many replicas of a partition must acknowledge the message before it is considered successfully written: 0 requiring no acks, 1 meaning only the leader broker and all, waiting for the complete set of replicas to acknowledge the record (CONFLUENT, 2024b).

- When "acks=0", the producer does not require acknowledgments, which is suitable for messages whose loss will not impact business operations. In this scenario, high availability is not a concern as the production or consumption of records is not critical.
- "acks=1" implies that the broker will acknowledge the producer once it has added the records to the log, irrespective of whether the other replicating brokers have done so. Acknowledged records can still be lost due to a broker crash or network partition. For instance, if the leader crashes immediately after appending locally but before the followers have received the appended records, the new leader would

not have received the acknowledged records.

- “acks=all” signifies that the leader will only send an acknowledgment when all ISR brokers have added the records to their log.

Each replica has one server acting as leader and others as followers. The leader handles the read/write while followers replicate the data. If the leader fails, any of the followers in the ISR is elected as the leader. A record of its topic is sent to the leader when a producer publishes it. The leader adds the record to its log and updates its offset. Only committed records are visible to consumers, and new data is stacked on the cluster. A producer needs to know the partition for each record, which can be specified by a key or chosen by the producer. The producer requests cluster metadata from the broker before sending any records. The metadata shows the leader for each partition, and the producer writes to the leader. The producer can use the key’s hash or its own logic to determine the partition (RAWAL, 2020).

One of the pitfalls of publishing records to Kafka is using **the same or null** key for all records, which causes all records to go to the same partition, leading to an unbalanced topic. The key determines the partition for each record by its hash or the producer’s logic. A balanced topic has records distributed evenly across partitions, which improves performance and scalability (CORTEZ, 2022). In short, both concepts can be summarized as:

- Partition: each topic can be split up into partitions for load balancing (you could write into different partitions at the same time) & scalability (the topic can scale up without the instance limitations); within the same partition, the records are ordered; A topic consists of a bunch of buckets. Each such bucket is called a partition. Kafka takes its hash and appends it into the appropriate bucket when publishing an item.
- Replica: for fault-tolerant durability mainly; This is the number of copies of topic data to be replicated across the network.

6.2.2 How consumers acquire records

The high-level consumer (the consumer group) consists of one or more consumers. A consumer group is created by adding the property "group.id" to a consumer. Giving the same group ID to another consumer means it will join the same group. The broker will distribute according to which consumer should read from which partitions and keep track

of which offset the group is at for each partition. It tracks this by having all consumers commit which offset they have handled. The consumption is rebalanced when a consumer is added or removed from a group. All consumers are stopped on every rebalance, so clients that time out or are restarted often will decrease the throughput.

Consumers pull messages from topic partitions. Different consumers can be responsible for different partitions. Kafka can support many consumers and retain large amounts of data with little overhead. By using consumer groups, consumers can be parallelized so that multiple consumers can read from multiple partitions on a topic, allowing a very high message processing throughput. The number of partitions impacts the maximum parallelism of consumers, as there cannot be more consumers than partitions. Records are never **pushed out** to consumers: the consumer will ask for messages when the consumer is ready to handle the message.

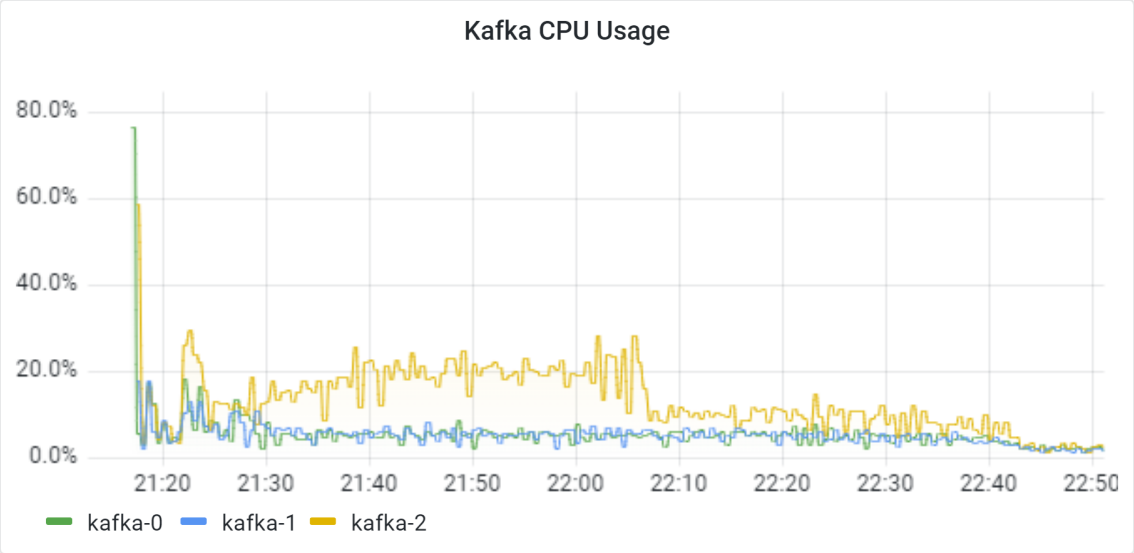
One experiment ran with 1 EMQX broker, 15 clients (which scale up to 225 simultaneous producers at peak), and 3 Kafka brokers without consumer grouping or replication. The problem in this scenario is the amount of different topics and the sheer amount of data generated while not controlling the partitioning inside the Kafka cluster. In Fig. 6.1, the cluster controller *kafka-0* is the one that does most of the computation, while the other two brokers barely compute. In Fig. 6.2, you can see that *kafka-2* got more topics sorted into it, storing more data and scaling memory usage until the maximum limit of memory in the first 20 minutes of the 90-minute experiment (the brokers were limited to a limit of 2 GB of memory per instance), while the other brokers do not come close to reaching it.

In another experiment, now while performing a **manual** distribution of topics pre-experiment run and executing with 5 Kafka brokers, the allocation of computation and memory did not scale exponentially and quickly bottleneck and were more distributed, but still being unable to split the brokers to use an even share of resources.

6.2.3 Kafka Producer Partitioner

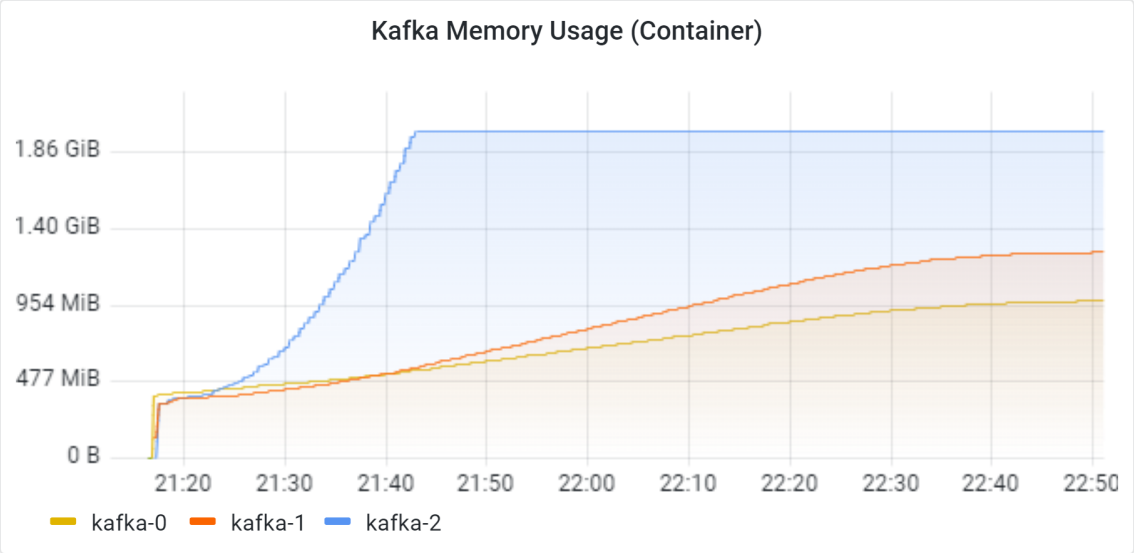
The partitioner is a component that determines how messages are assigned to partitions in a Kafka topic, where the message will physically be relocated. The partitioner can be either the default one provided by Kafka or a custom one implemented by the producer. The partitioner takes the message key, a parameter sent alongside the message value, and the number of partitions as inputs and returns the partition ID as output. The partitioner can use different algorithms, such as hashing, round-robin, or range partition-

Figure 6.1 – Topic distribution scenario 1: Kafka CPU Usage



Source: The Author

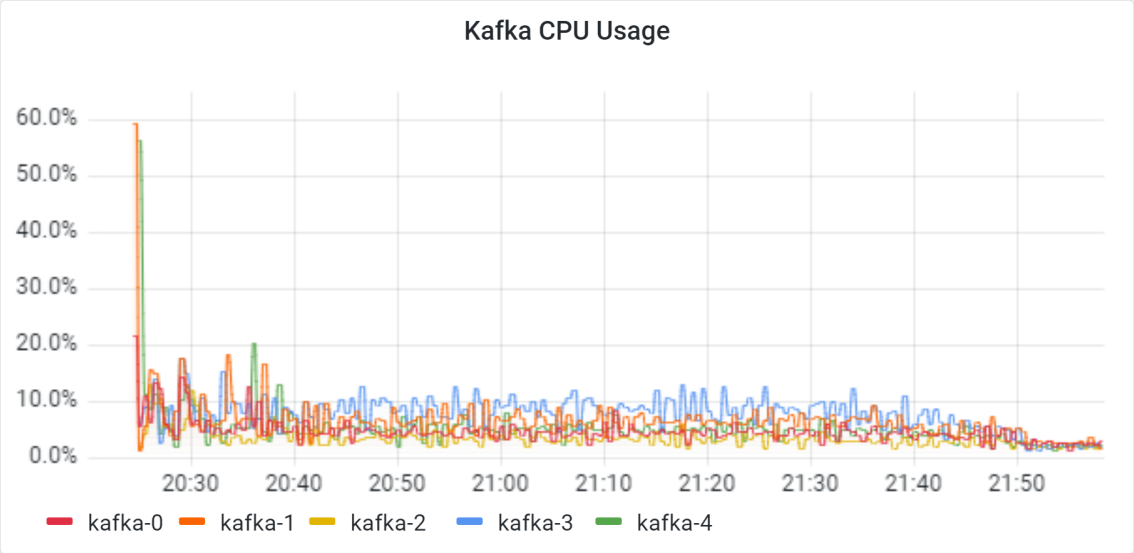
Figure 6.2 – Topic distribution scenario 1: Kafka Memory Usage



Source: The Author

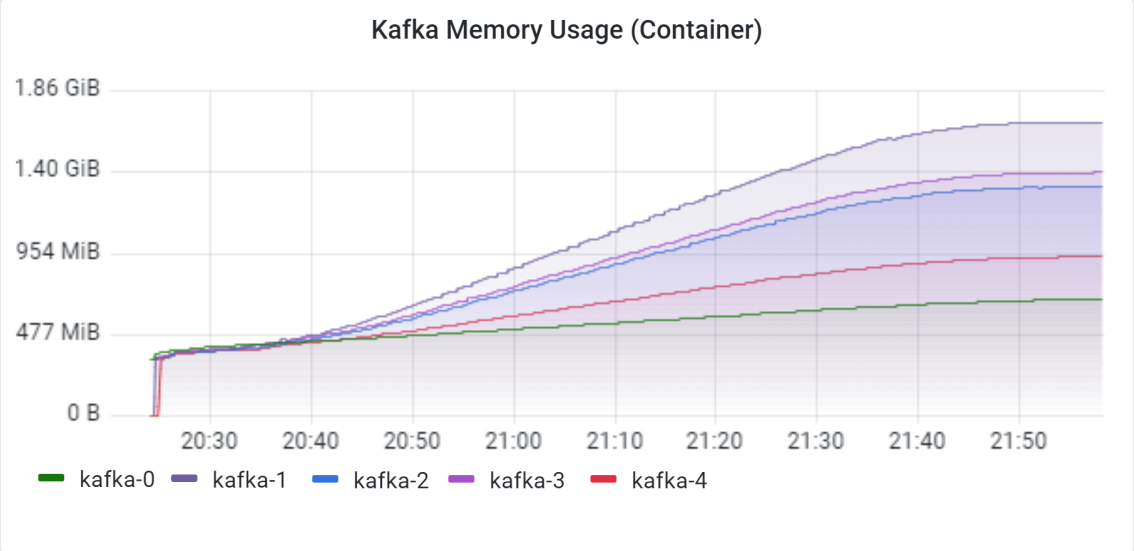
ing, to map keys to partitions. The choice of partitioner affects the ordering, scalability, and performance of the Kafka system. Kafka has different behaviors defined in case the key is and is not defined. When the key is not null, key hashing is the process used. By default, the Kafka partitioner uses MurmurHash2, a very low latency non-cryptographic hashing algorithm suitable for hash-based lookup (APPLEBY, 2010). Using the same key will redirect messages towards the same partition. The default partitioning strategy is:

Figure 6.3 – Topic distribution scenario 2: Kafka CPU Usage



Source: The Author

Figure 6.4 – Topic distribution scenario 2: Kafka Memory Usage



Source: The Author

- If a partition is specified in the record, use it;
- If no partition is specified, but there is a key present, generate a murmur2 hash from that key and compare it to the partition count to assign it to a partition;
- If there is no partition or key, use the sticky partitioner.

When there is no partition and no key specified, it is widely believed that Kafka partitions in a round-robin fashion. That information is false as of Kafka 2.4, and Kafka

implements a new method called the Sticky Partitioner. Versions previous to choose partitions based on round-robin if no partition or key was defined. Not defining partitions or keys and evenly distributing the records among the partitions creates more and smaller batches. It might be better to send all the records to one partition (or a few partitions) in a bigger batch.

The sticky partitioner implementation solves the issue of dividing keyless records into smaller groups by selecting one partition for all such records. When that partition's group is full or done, the sticky partitioner randomly picks and stays with another partition. This ensures that over time, records are roughly balanced among all the partitions and also benefit from larger batch sizes [1]. This algorithm leads to larger batches with reduced latency. Over time, records will be evenly spread, so the balance of the cluster is not affected, improving the producer's performance. The batch size directly impacts latency from the producer to the broker. Bigger batches mean fewer requests and less queuing, which lowers latency. This is true even when the "linger.ms" configuration is zero. When "linger.ms" is on, low throughput can increase latency if the batches are not full enough to be sent before "linger.ms". Increasing the batch size to send them sooner can reduce latency further. The sticky partitioner tries this by staying with a partition until the batch is full (or sent when "linger.ms" is over), which creates bigger batches and lowers latency than the default partitioner. Even when "linger.ms" is zero and we send immediately, we get better batching and lower latency (OLSHAN, 2019).

6.3 Choosing Topic Replication Factor and Partition Count

A partition count and the replication factor are required when creating a new topic in Kafka. Starting with one set of values and changing them later can have a meaningful impact on system performance. Kafka stores data by topic, and each topic comprises several partitions representing a logical slice, or piece, of the records on that topic. Partitions may be replicated across brokers, creating a copy of each record sent to a partition and physically stored in a log on multiple different brokers. The copies of the log are called replicas. To balance the replication factor of a Kafka topic, it is crucial to replicate partitions to enhance system reliability and fault tolerance. However, it is essential to note that

[1] More information on the sticky partitioner can be found at `https://conduktor.io/kafka/producer-default-partitioner-and-sticky-partitioner` and `https://confluent.io/blog/apache-kafka-producer-improvements-sticky-partitioner`.

replicating topics does not boost consumer parallelism. When deciding on a replication factor, the ideal range should be between 2 and 4, achieving an optimal performance and fault tolerance balance. For that reason, cloud providers offer 3 data center availability zones within a single region. A higher replication factor contributes to the resilience of the system. With a replication factor denoted as “N”, the system can withstand the failure of up to “N-1” brokers without affecting availability if acks are set to “0” or “1”. If acks are set to “all”, the system can tolerate the failure of N minus the number of brokers set in the “min.insync.replicas” configuration.

Having a high replication factor while providing more resilience also presents its own set of disadvantages. As data needs to be replicated across all brokers if acks=all, this causes a higher latency for the producers to push their data while also requiring additional physical storage in the system to allocate the replicated partitions. The guideline is to follow the rule of threes: start with three partitions. Having only one means having no fault tolerance, meaning that the data stored is gone if the partition is lost. Having three partitions replicated means that if one fails, fault tolerance is never lost unless a critical failure affects them all, but that is why cloud providers offer the use of systems separated geographically, reducing the chances of simultaneous failure. A cluster of three controllers can withstand a single failure. Similarly, a cluster with five controllers can endure two failures, and the pattern continues in this manner. The number of in-sync replicas should be greater than one for the same reason: the ISR always has at least two working replicas, and the producer ack configuration demands all partitions to acknowledge it.

6.4 Partitioning Kafka in practice

Kafka offers several functionalities for achieving data consistency and durability and improving communication resilience. As with any architectural decision, it will be up to the developers of the digital twin to decide, based on its upsides and downsides, if injecting its usage will improve the overall system. One big worry is the computation wasted to support this type of system. Are we saving time by using it, or do we just have to dedicate more hardware that could be used to improve the start of the process, extracting data or the end goal of processing it instead of a middle-ground solution?

IoT faces a big problem with the growing complexity of code, and the more devices we want to connect to generate more data and compute even more complex models, the more complex the need for more computing and electrical resources than ever. Digi-

tal twin models can be remarkably complex, possessing several types of components and machinery, some of which are regularly changing, being upgraded, and expanded. Applications should be stateless, and usage of their services should not require being locked to specific technologies, while using cloud-native platforms that provide resilience and redundancy in case of failures and handle abstractions such as environmental changes and discovery of services (CORREA, 2022).

A digital twin using Kafka aims to achieve the lowest latency while still having a high throughput. Compression significantly increased the system's throughput without an increase in total latency. Reducing the distance between the brokers and the consumer clients reduced the latency, but the overall throughput did not change (FENNELL, 2022).

When employing so many technologies, as cited, for even more complex scenarios for the digital twin that is being constructed, one can argue that the focus on data optimization is not worth the extra time taken on just pure configuration. It is hard to individually map the distribution of all topics, especially with a growing data set. Rebalancing the cluster is another big topic. The wasted time in message queueing requires too many resources, which could be spent on acquiring more data or processing models to improve the digital twin.

When approaching how to model digital twins, one should focus on something other than technology and instead on the problem itself: data ingestion. How are we obtaining data, how can we use it to improve our model, and how fast can we do that? This is not a question that can not be universally solved. This is not a problem individualized to using Apache Kafka but a design decision that will affect the built architecture independent of the solution chosen. Damjanovic-Behrendt and Behrendt (DAMJANOVIC-BEHRENDT; BEHRENDT, 2019) present a microservices-based approach for constructing and planning digital twins, mapping out the underlying problems and suggesting several different open-source tools to tackle them. Requirements can significantly differ across industries and use cases. It is crucial to comprehend the objectives and the expenses to attain successful results. The cost can be evaluated in terms of both infrastructure requirements and operational intricacy. Each of these factors needs to be considered to achieve high performance.

7 CONCLUSION

This work presented a study on cloud and edge computing-enabled digital twins, digital replica systems capable of understanding and expanding on the functionalities of their real counterparts, exploring the challenges and opportunities of deploying and communicating with a digital twin system in a distributed environment. For digital twins to suggest actions, they need the power to command their physical counterparts, with a reliable and safe real-time link between the real and the virtual worlds. The main challenge of this study was to find ways to improve the performance of data-intensive systems with increasing data input while meeting time, cost, and complexity requirements. These systems belonged to the domain of digital twins - dynamic real-time systems that produce and process data. Digital twins depend on efficient data collection from their system parts and effective data management, considering their significant and ever-expanding data sources. This study investigated how data acquisition systems and event processing were related in the context of digital twins. Scaling these systems was difficult because moving the computation to the cloud could affect their real-time demands.

This work aimed to study how digital twin systems can scale up using cloud and edge computing, testing the real-time communication needs of a digital twin system deployed to a cloud and edge cluster with multiple components. This work was built on previous research in the field, and experiments were run with MQTT and Apache Kafka to check for possible issues for cloud and edge devices. The experiments used a real network configuration with a Kubernetes cluster and a mix of Raspberry Pis, desktop-grade computers, and server-grade machines, unlike earlier simulations that only used virtual machines, to get more realistic and accurate results. However, these experiments still did not fully mimic real-world situations. The experiments showed potential problems in the scenario runs caused by the brokers needing more resources than the edge devices with low specifications could provide. They discussed the feasibility of using event-processing and multi-broker systems with edge devices, considering the systems' features and suggesting possible solutions.

Experiment 1 presented a scenario of issues caused by using a single MQTT broker, which causes zero tolerance in case of a broker failure, losing all queued messages. EMQX can mitigate this using retained messages, which store the most recent message of a topic. However, this only solves some of the problems and forces the need for data replication through multiple broker nodes.

Experiment 2 then furthers this by clustering the broker into multiple connected nodes. However, the existing cluster implementation performs a full mesh connection, with all nodes storing the subscribed topics from each node and forwarding received messages to that node if needed. The scenario in this experiment shows an example of what happens when randomly distributing where the clients will publish and where they will subscribe. It features a long tail of extra time of just one broker pushing messages to another broker, adding another hour until all messages get sent to the subscribers. Possible fixes could be protocol-bound, forcing reconnection to the direct broker, either from the publisher or the subscriber, reducing broker retransmission that causes additional traffic, and impacting the other topics being processed in the same nodes. More recent versions of EMQX now feature a new broker topology, with only core nodes storing the broker topics in a full mesh with other core nodes. In contrast, replicant nodes connect to one core node and the clients, reducing the retransmissions needed in the cluster.

Experiment 3 focused on the performance of the Apache Kafka node cluster, presenting the new quorum protocol used by Kafka for self-managing metadata instead of relying on the ZooKeeper external dependency. The scenarios presented show problems with high memory usage and computation distribution between nodes and argue that despite the several benefits to communication, adding a heavy application like Kafka to the architecture might not always benefit the end user. Performing a manual distribution of topics, knowing where topics will be, and providing a direct link to them, instead of relying on retransmissions inside the cluster, alleviates the bottlenecks in memory and CPU, supporting an increased message rate, but that requires manual intervention in the system.

The experiments also discussed the trade-offs and limitations of using Kafka for digital twins, specifically in these low-specification scenarios. They suggested alternatives and improvements for future work, considering the technologies used. Analysis of the system's architecture is critical to understanding the real requirements to run a system like this. Depending on how critical the execution is, the system must operate independently of a central cloud without causing a failure in operations. Adding an extra computation layer to a very data-heavy application might overload the hardware accessible by the digital twin system, which might not have access to external data centers to offload computation. Kafka configuration is also a complex matter, especially for non-specialist users, and requires maintenance to handle correct topic partitioning, data acquisition, and replication factors.

Using many technologies, as mentioned, for more complex digital twin scenarios,

one may question the value of data optimization over the time spent on configuration. It is challenging to map the distribution of all topics with a growing data set. Rebalancing the cluster is another issue. The time lost in message queueing wastes resources that could be used for more data or model improvement. Implementations by the industry are moving their focus on using services provided by the major cloud providers, already embedded into their platforms, instead of individual, specialized software. While this offers ease of usage, it limits the amount of customization and functionality of the digital twin. The main problem for digital twin modeling is data ingestion: how to get data, how to improve the simulation model, and how fast we can do it so the digital twin can provide benefits. Understanding the goals and costs to achieve good outcomes is vital. Both infrastructure and operational complexity can affect the cost and needs to be handled case-by-case. The feasibility and benefits of using cloud and edge computing for digital twins are a big topic in academia, as well as the need for further research and optimization to address such systems' real-time requirements and data management issues, allowing for their usage in the context of edge computing.

REFERENCES

3GPP. **Policy and charging control architecture**. [*S. l.*], dez. 2021. Version 17.2.0. Disponível em: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=810.

ALAM, Kazi Masudul; EL SADDIK, Abdulmotaleb. C2PS: A Digital Twin Architecture Reference Model for the Cloud-Based Cyber-Physical Systems. **IEEE Access**, PP, p. 1–1, jan. 2017. DOI: 10.1109/ACCESS.2017.2657006.

ALONSO, Rubén; LOCCI, Riccardo; REFORGIATO RECUPERO, Diego. Improving digital twin experience through big data, IoT and social analysis: An architecture and a case study. **Heliyon**, v. 10, n. 2, e24741, 2024. ISSN 2405-8440. DOI: https://doi.org/10.1016/j.heliyon.2024.e24741.

APPLEBY, Austin. MurmurHash2, nov. 2010. Disponível em: https://github.com/aappleby/smhasher/blob/master/src/MurmurHash2.cpp. Acesso em: 28 jan. 2024.

ASLAMOVA, Yulia. **Kubernetes in simple words: explained by Eric Swildens**. 2021. Disponível em: https://web.archive.org/web/20221231160308/https://nimbella.com/blog/kubernetes-in-simple-words-explained-by-eric-swildens. Acesso em: 4 mar. 2023.

ATLAM, Hany F; WALTERS, Robert J; WILLS, Gary B. Fog Computing and the Internet of Things: A Review. **Big Data and Cognitive Computing**, Multidisciplinary Digital Publishing Institute, v. 2, n. 2, p. 10, 2018.

ATMOKO, R.A.; RIANTINI, R.; HASIN, M.K. IoT real time data acquisition using MQTT protocol. *In* IOP PUBLISHING, 1. JOURNAL of Physics: Conference Series. [*S. l.: s. n.*], 2017. v. 853, p. 012003.

BANKS, Andrew; BRIGGS, Ed; BORGENDALE, Ken; GUPTA, Rahul. **MQTT Version 5.0**. Mar. 2019. Disponível em: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html. Acesso em: 6 out. 2022.

BARR, Jeff. **AWS IoT – Cloud Services for Connected Devices**. 2015. Disponível em: https://aws.amazon.com/blogs/aws/aws-iot-cloud-services-for-connected-devices/. Acesso em: 30 out. 2020.

BARROS, Ândreo Dias. **A Study on the Integration of Data Acquisition and Processing for Digital Twins with Kafka and MQTT**. 2022. Bachelor's thesis – Federal University of Rio Grande do Sul.

BEJECK, Bill. **Kafka Streams in Action: Real-time apps and microservices with the Kafka Streams API**. [*S. l.*]: Simon e Schuster, 2018. ISBN 9781617294471. Disponível em: https://books.google.com.br/books?id=RTgzEAAAQBAJ.

BENTLEY. **Digital Twins**. [*S. l.*], 2023. Disponível em: https://www.bentley.com/software/digital-twins/. Acesso em: 21 mai. 2023.

BLIZZARD. **node-rdkafka - Node.js wrapper for Kafka C/C++ library**. [*S. l.*], 2023. Disponível em: https://github.com/Blizzard/node-rdkafka.

BOSCHERT, Stefan; ROSEN, Roland. Digital Twin—The Simulation Aspect. *In*. MECHATRONIC Futures. [*S. l.*]: Springer, 2016. p. 59–74.

CAN, Ozgu; TURKMEN, Aytug. Digital Twin and Manufacturing. **Digital Twin Driven Intelligent Systems and Emerging Metaverse**, Springer, p. 175–194, 2023.

CARDIN, Olivier. Classification of cyber-physical production systems applications: Proposition of an analysis framework. **Computers in Industry**, Elsevier, v. 104, p. 11–21, 2019.

CHANDRAKANT, Kumar. Kafka's Shift from ZooKeeper to Kraft, fev. 2024. Disponível em: https://web.archive.org/web/20240126131156/https://www.cloverdx.com/tech-blog/how-to-connect-and-publish-messages-to-kafka. Acesso em: 18 fev. 2024.

CHEN, Yishan; DENG, Shuiguang; MA, Hongtao; YIN, Jianwei. Deploying data-intensive applications with multiple services components on edge. **Mobile Networks and Applications**, Springer, p. 1–16, 2019.

CONFLUENT. **Confluent Platform System Requirements**. [*S. l.*], 2024. Disponível em: https://web.archive.org/web/20240130035711/https://docs.confluent.io/platform/current/installation/system-requirements.html.

CONFLUENT. **Kafka Producer Configurations**. [*S. l.*], 2024. Disponível em: https://docs.confluent.io/platform/current/installation/configuration/producer-configs.html.

CORREA, Gustavo de Medeiros. **A cloud, microservice-based digital twin for the Oil industry a flow assurance case study**. 2022. Bachelor's thesis – Federal University of Rio Grande do Sul. DOI: 10183/254401.

CORTEZ, Calvin. How to connect and publish messages to Kafka, fev. 2022. Disponível em: https://web.archive.org/web/20240126131156/https://www.cloverdx.com/tech-blog/how-to-connect-and-publish-messages-to-kafka. Acesso em: 4 fev. 2022.

D'SILVA, Godson Michael; KHAN, Azharuddin; GAURAV; BARI, Siddhesh. Real-time processing of IoT events with historic data using Apache Kafka and Apache Spark with dashing framework. *In*. 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT). [*S. l.: s. n.*], 2017. p. 1804–1809. DOI: 10.1109/RTEICT.2017.8256910.

DAMJANOVIC-BEHRENDT, Violeta; BEHRENDT, Wernher. An open source approach to the design and implementation of Digital Twins for Smart Manufacturing. **International Journal of Computer Integrated Manufacturing**, Taylor & Francis, v. 32, n. 4-5, p. 366–384, 2019.

DENG, Jing; HUANG, Scott C-H; HAN, Yunghsiang S; DENG, Julia H. Fault-tolerant and reliable computation in cloud computing. *In*. 2010 IEEE Globecom Workshops. Miami, FL, USA: IEEE, 2010. p. 1601–1605.

DIZDAREVIĆ, Jasenka; CARPIO, Francisco; JUKAN, Admela; MASIP-BRUIN, Xavi. A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration. **ACM Computing Surveys**, Association for Computing Machinery, New York, NY, USA, v. 51, n. 6, 2019.

EMQ INC. **EMQX**. 2023. Disponível em: https://web.archive.org/web/20230313020238/https://www.emqx.io/docs/en/v5.0/. Acesso em: 12 mar. 2023.

EMQ INC. **EMQX Clustering**. 2024. Disponível em: https://web.archive.org/web/20240209183154/https://www.emqx.io/docs/en/latest/design/clustering.html. Acesso em: 9 fev. 2024.

EMQ INC. Reaching 100M MQTT connections with EMQX 5.0, mai. 2022. Disponível em: https://web.archive.org/web/20240219045912/https://www.emqx.com/en/blog/reaching-100m-mqtt-connections-with-emqx-5-0. Acesso em: 18 fev. 2024.

FENNELL, Conor. **A Communication Architecture for Transportation Digital Twins using Apache Kafka**. 2022. Diss. (Mestrado) – University of Dublin.

FERRARI, Paolo; SISINNI, Emiliano; BRANDÃO, Dennis; ROCHA, M. Evaluation of communication latency in industrial IoT applications. *In* IEEE. 2017 IEEE International Workshop on Measurement and Networking (M&N). [*S. l.: s. n.*], 2017. p. 1–6.

GLUSHACH, Roman. The Evolution of Kafka Architecture: From ZooKeeper to KRaft, jul. 2022. Disponível em: https://web.archive.org/web/20240219162748/https://romanglushach.medium.com/the-evolution-of-kafka-architecture-from-zookeeper-to-kraft-f42d511ba242. Acesso em: 19 fev. 2024.

GRGIĆ, Krešimir; ŠPEH, Ivan; HEĐI, Ivan. A web-based IoT solution for monitoring data using MQTT protocol. *In* IEEE. 2016 international conference on smart systems and technologies (SST). [*S. l.: s. n.*], 2016. p. 249–253.

GRIEVES, Michael. **Origins of the Digital Twin Concept**. [*S. l.*], 2016.

GRIEVES, Michael; VICKERS, John. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. *In*. **Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches**. Edição: Franz-Josef Kahlen, Shannon Flumerfelt e Anabela Alves. [*S. l.*]: Springer International Publishing, 2017. p. 85–113.

GUTHRIE, Georgina. **A simple introduction to Kubernetes and the world of containers**. 2022. Disponível em: https://web.archive.org/web/20230305021450/https://nulab.com/learn/software-development/introduction-kubernetes-containers/. Acesso em: 4 mar. 2023.

HALL, Lena. **How to process streams of data with Apache Kafka and Spark**. [*S. l.*], jun. 2018. Disponível em: https://web.archive.org/web/20230319192444/https://cloudblogs.microsoft.com/opensource/2018/07/09/how-to-data-processing-apache-kafka-spark/.

HUMAN, C.; BASSON, A. H.; KRUGER, K. Digital Twin Data Pipeline Using MQTT in SLADTA. *In* BORANGIU, Theodor; TRENTESAUX, Damien; LEITÃO, Paulo; CARDIN, Olivier; LAMOURI, Samir (Ed.). **Service Oriented, Holonic and Multi-Agent Manufacturing Systems for Industry of the Future**. Cham: Springer International Publishing, 2021. p. 111–122. ISBN 978-3-030-69373-2.

IBM. **IoT and IBM Cloud are creating opportunities for growth**. [*S. l.*], 2023. Disponível em: https://www.ibm.com/cloud/internet-of-things. Acesso em: 21 mai. 2023.

JACOBY, Michael; USLÄNDER, Thomas. Digital Twin and Internet of Things—Current Standards Landscape. **Applied Sciences**, Multidisciplinary Digital Publishing Institute, v. 10, n. 18, p. 6519, 2020.

JEFFERY, Andrew; HOWARD, Heidi; MORTIER, Richard. Rearchitecting Kubernetes for the Edge. *In*. PROCEEDINGS OF THE 4TH INTERNATIONAL WORKSHOP ON EDGE SYSTEMS, ANALYTICS AND NETWORKING. Online, United Kingdom: Association for Computing Machinery, 2021. (EdgeSys '21), p. 7–12. ISBN 9781450382915. DOI: 10.1145/3434770.3459730. Disponível em: https://doi.org/10.1145/3434770.3459730.

JOSEPH, Christina Terese; CHANDRASEKARAN, K. Straddling the crevasse: A review of microservice software architecture foundations and recent advancements. **Software: Practice and Experience**, Wiley Online Library, v. 49, n. 10, p. 1448–1484, 2019.

KHAN, WZ; REHMAN, MH; ZANGOTI, HM; AFZAL, MK; ARMI, N; SALAH, K. Industrial internet of things: Recent advances, enabling technologies and open challenges. **Computers & Electrical Engineering**, Elsevier, v. 81, p. 106522, 2020.

KNEBEL, Francisco Paiva. **An open Digital Twin framework based on microservices in the cloud**. 2020. Bachelor's thesis – Federal University of Rio Grande do Sul. DOI: 10183/219160.

KNEBEL, Francisco Paiva; TREVISAN, Rafael; NASCIMENTO, Givanildo Santana do; ABEL, Mara; WICKBOLDT, Juliano Araujo. A study on cloud and edge computing for the implementation of digital twins in the Oil & Gas industries. **Computers & Industrial Engineering**, v. 182, p. 109363, 2023. ISSN 0360-8352. DOI: https://doi.org/10.1016/j.cie.2023.109363. Disponível em: https://www.sciencedirect.com/science/article/pii/S036083522300387X.

KNEBEL, Francisco Paiva; WICKBOLDT, Juliano Araujo; FREITAS, Edison Pignaton de. A Cloud-Fog Computing Architecture for Real-Time Digital Twins. **arXiv preprint arXiv:2012.06118**, 2020. Disponível em: https://arxiv.org/abs/2012.06118.

KOZIOLEK, Heiko; GRÜNER, Sten; RÜCKERT, Julius. A Comparison of MQTT Brokers for Distributed IoT Edge Computing. *In* SPRINGER. EUROPEAN Conference on Software Architecture. [*S. l.: s. n.*], 2020. p. 352–368.

KRITZINGER, Werner; KARNER, Matthias; TRAAR, Georg; HENJES, Jan; SIHN, Wilfried. Digital Twin in manufacturing: A categorical literature review and classification. **IFAC-PapersOnLine**, v. 51, n. 11, p. 1016–1022, 2018. 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018. ISSN 2405-8963. DOI: 10.1016/j.ifacol.2018.08.474.

KUBERNETES. **Kubernetes Documentation / Concepts / Overview**. 2023. Disponível em: https://kubernetes.io/docs/concepts/overview. Acesso em: 4 mar. 2023.

LARSSON, Rasmus. **Creating Digital Twin Distributed Networks Using Switches With Programmable Data Plane**. [*S. l.: s. n.*], 2021.

LINUX FOUNDATION. **A simple introduction to Kubernetes and the world of containers**. 2023. Disponível em: https://web.archive.org/web/20230305030010/https://www.cncf.io/reports/cncf-annual-survey-2022/. Acesso em: 4 mar. 2023.

LÓPEZ, Carlos Eduardo Belman. Real-time event-based platform for the development of digital twin applications. **The International Journal of Advanced Manufacturing Technology**, v. 116, n. 3, p. 835–845, set. 2021. ISSN 1433-3015. DOI: 10.1007/s00170-021-07490-9. Disponível em: https://doi.org/10.1007/s00170-021-07490-9.

MAALOUL, Sassi; ANISS, Hasnaa; KASSAB, Mohamed; BERBINEAU, Marion. Classification of C-ITS Services in Vehicular Environments. **IEEE Access**, v. 9, p. 117868–117879, ago. 2021. DOI: 10.1109/ACCESS.2021.3105815.

MARINESCU, Dan C. **Cloud computing: theory and practice**. [*S. l.*]: Morgan Kaufmann, 2017.

MCCABE, Colin. Apache Kafka Needs No Keeper: Removing the Apache ZooKeeper Dependency, mai. 2020. Disponível em: https://web.archive.org/web/20240218201638/https://www.confluent.io/blog/removing-zookeeper-dependency-in-kafka/. Acesso em: 18 fev. 2024.

MELL, Peter; GRANCE, Tim *et al.* The NIST definition of cloud computing. **National Institute of Science and Technology, Special Publication, 800**, Computer Security Division, Information Technology Laboratory, National Institute of Standards e Technology, v. 145, 2011.

MICROSOFT. **Azure Digital Twins**. [*S. l.*], 2023. Disponível em: https://azure.microsoft.com/en-us/products/digital-twins.

MICROSOFT. **Communicate with your IoT hub using the MQTT protocol**. 2018. Disponível em: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-mqtt-support. Acesso em: 30 out. 2020.

MICROSOFT. **What is Azure Digital Twins?** [*S. l.*], 2023. Disponível em: https://learn.microsoft.com/en-us/azure/digital-twins/overview.

OKEKE, Franklin. **AWS vs Azure for Industrial IoT: Which solution is best for your business in 2023?** [*S. l.*], 2022. Disponível em: https://www.techrepublic.com/article/aws-vs-azure-industrial-iot/.

OLSHAN, Justine. KIP-480: Sticky Partitioner, jun. 2019. Disponível em: https://cwiki.apache.org/confluence/display/KAFKA/KIP-480%3A+Sticky+Partitioner. Acesso em: 28 jan. 2024.

QI, Qinglin; TAO, Fei; HU, Tianliang; ANWER, Nabil; LIU, Ang; WEI, Yongli; WANG, Lihui; NEE, Andrew. Enabling technologies and tools for digital twin. **Journal of Manufacturing Systems**, v. 58, p. 3–21, mar. 2021. DOI: 10.1016/j.jmsy.2019.10.001.

RATHI, Kunal. **Kafka Streaming Vs Spark Streaming**. [*S. l.*], 2023. Disponível em: https://azureops.org/articles/kafka-streaming-vs-spark-streaming/.

RAWAL, Kajal. Kafka Broker, Kafka Topic, Consumer and Record Flow in Kafka, set. 2020. Disponível em: https://web.archive.org/web/20240126130345/https://kajalrawal.medium.com/kafka-broker-kafka-topic-consumer-and-record-flow-in-kafka-ec55104977b8. Acesso em: 22 set. 2020.

RUBENSTONE, Jeff. Bentley Launches Digital Twin Tools, Infrastructure Cloud, 2023. Disponível em: https://www.enr.com/articles/55445-bentley-launches-digital-twin-tools-infrastructure-cloud. Acesso em: 21 mai. 2023.

SHANKAR, Ravi; GVK, Sasirekha; RAMANATHAN, Chandrashekar; BAPAT, Jyotsna. Knowledge-based Digital Twin for Oil and Gas 4.0 Upstream Process: A System Prototype. *In*. 2022 IEEE INTERNATIONAL CONFERENCE ON INTERNET OF THINGS AND INTELLIGENCE SYSTEMS (IOTAIS). [*S. l.: s. n.*], 2022. p. 344–350. DOI: 10.1109/IoTaIS56727.2022.9975974.

SIEMENS. Enhancing efficiency and developing new revenue streams with Insights Hub, mai. 2023. Disponível em: https://web.archive.org/web/20230521224052/https://resources.sw.siemens.com/en-US/case-study-orisol. Acesso em: 21 mai. 2023.

SIEMENS. MindSphere, mar. 2023. Disponível em: https://web.archive.org/web/20230308180557/https://www.plm.automation.siemens.com/global/en/products/mindsphere/. Acesso em: 8 mar. 2023.

SOUSA, Adriano B de; NETO, Jose R Torres; GERALDO FILHO, PR; UEYAMA, Jo. Uma plataforma de IoT para integração de dispositivos baseada em nuvem com Apache Kafka. *In* SBC. ANAIS Estendidos do XXXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. Porto Alegre, RS, Brasil: SBC, 2018. DOI: 10.5753/sbrc_estendido.2018.14638. Disponível em: https://sol.sbc.org.br/index.php/sbrc_estendido/article/view/14638.

TAO, Fei; SUI, Fangyuan; LIU, Ang; QI, Qinglin; ZHANG, Meng; SONG, Boyang; GUO, Zirong; LU, Stephen; NEE, Andrew. Digital twin-driven product design framework. **International Journal of Production Research**, v. 57, p. 1–19, fev. 2018. DOI: 10.1080/00207543.2018.1443229.

TAO, Fei; ZHANG, Meng; LIU, Yushan; NEE, A.Y.C. Digital twin driven prognostics and health management for complex equipment. **CIRP Annals**, v. 67, n. 1, p. 169–172, 2018. ISSN 0007-8506. DOI: https://doi.org/10.1016/j.cirp.2018.04.055. Disponível em: https://www.sciencedirect.com/science/article/pii/S0007850618300799.

TAO, Fei; ZHANG, Meng; NEE, Andrew. Digital Twin and Cloud, Fog, Edge Computing. *In*. DIGITAL TWIN DRIVEN SMART MANUFACTURING. [*S. l.: s. n.*], 2019. p. 171–181. ISBN 9780128176306. DOI: 10.1016/B978-0-12-817630-6.00008-4.

TREVISAN, Rafael; KNEBEL, Francisco; WICKBOLDT, Juliano. Uma Avaliação do uso de MQTT para a Implementação de Digital Twins. *In*. ANAIS da XVIII Escola Regional de Redes de Computadores. Evento Online: SBC, 2020. p. 41–47. DOI: 10.5753/errc.2020.15187. Disponível em: https://sol.sbc.org.br/index.php/errc/article/view/15187.

TREVISAN, Rafael; KNEBEL, Francisco; WICKBOLDT, Juliano; ABEL, Mara. Aquisição de Dados Escalável e Ciente da Aplicação para Gêmeos Digitais. *In*. ANAIS DO XXVII WORKSHOP DE GERÊNCIA E OPERAÇÃO DE REDES E SERVIÇOS. Fortaleza: SBC, 2022. p. 57–70. DOI: 10.5753/wgrs.2022.223478. Disponível

em:
https://sol.sbc.org.br/index.php/wgrs/article/view/21477.

WARE, Kary. **MQTT Broker Cluster Scalability: How is it done in EMQX?** [*S. l.: s. n.*], abr. 2022. Disponível em: https://assets.emqx.com/resources/ebooks/mqtt-broker-cluster-scalability.pdf. Acesso em: 18 fev. 2024.

XIONG, Minglan; WANG, Huawei. Digital twin applications in aviation industry: A review. **The International Journal of Advanced Manufacturing Technology**, v. 121, n. 9, p. 5677–5692, ago. 2022. ISSN 1433-3015. DOI: 10.1007/s00170-022-09717-9. Disponível em: https://doi.org/10.1007/s00170-022-09717-9.

ZHANG, Lucy. **Building Facebook Messenger**. 2011. Disponível em: https://www.facebook.com/notes/facebook-engineering/building-facebook-messenger/10150259350998920. Acesso em: 30 out. 2020.

ZHANG, Zixuan; WEN, Feng; SUN, Zhongda; GUO, Xinge; HE, Tianyiyi; LEE, Chengkuo. Artificial Intelligence-Enabled Sensing Technologies in the 5G/Internet of Things Era: From Virtual Reality/Augmented Reality to the Digital Twin. **Advanced Intelligent Systems**, v. 4, n. 7, p. 2100228, 2022. DOI: https://doi.org/10.1002/aisy.202100228.

APPENDIX A — RESUMO EXPANDIDO (PORTUGUESE)

Resolução 02/2021 – Redação de Teses e Dissertações em Inglês Dissertações de Mestrado e Teses de Doutorado do PPGC, bem como outros trabalhos escritos tais como Proposta de Tese e PEP, poderão ser redigidas em inglês desde que contenham um título e resumo expandido redigidos em português. O resumo expandido deve conter no mínimo duas páginas inteiras, deve aparecer como apêndice e deve conter as principais contribuições e resultados do trabalho.

Projetando e Implementando Gêmeos Digitais com Computação em Nuvem e de Borda: Desafios e Oportunidades

INTRODUÇÃO: Gêmeos digitais são representações interconectadas de um sistema físico e um virtual que se comunicam em tempo real. O elemento virtual é um software que reflete as ações do seu correspondente físico, identificando problemas e solucionando-os de maneira autônoma, sem a necessidade de intervenção humana. Com o surgimento da Internet das Coisas e de dispositivos sensoriais de baixo custo, nasceu o gêmeo digital moderno, que precisa ser capaz de controlar suas contrapartes físicas, exigindo uma conexão confiável. Esta pesquisa explora a relação entre sistemas de coleta de dados e processamento de eventos no contexto de gêmeos digitais, utilizando computação em nuvem e de borda. O principal desafio desta pesquisa é desenvolver métodos para melhorar o desempenho de sistemas intensivos em dados com entrada de dados crescente, respeitando as restrições de tempo, custo e complexidade. Este estudo apresenta uma análise sobre gêmeos digitais que utilizam computação em nuvem e de borda, experimentando com comunicação em tempo real de um sistema de gêmeos digitais implementado em um ambiente de nuvem multicamadas e borda. Este trabalho expande o escopo para projetar sistemas de gêmeos digitais para operar em escala, realizando experimentos usando MQTT e Apache Kafka para identificar possíveis problemas para dispositivos de nuvem e de borda. Os experimentos revelaram possíveis problemas nas execuções dos cenários devido ao fato de os corretores exigirem mais recursos do que os dispositivos de borda de baixa especificação poderiam oferecer e, ao mesmo tempo, discutindo os compromissos e as restrições do uso do Kafka para gêmeos digitais, principalmente em cenários de baixa especificação. A introdução de uma camada de computação adicional em um aplicativo com uso intenso de dados pode sobrecarregar o hardware disponível para o sistema de gêmeos digitais. A viabilidade e os benefícios de usar a computação em nuvem e de borda para gêmeos digitais são tópicos acadêmicos importantes. Há também a necessidade de mais pesquisas e otimização para atender aos requisitos de tempo real e aos problemas de gerenciamento de dados desses sistemas, permitindo sua aplicação na computação de borda.

FUNDAMENTAÇÃO TEÓRICA: Computação Distribuída pode processar os grandes volumes de dados gerados pelos gêmeos digitais, permitindo a análise e o monitoramento em tempo real de ativos e sistemas físicos. Isso envolve o uso de bancos de dados distribuídos, computação em nuvem, computação de borda e outras tecnologias para dis-

tribuir a carga de trabalho e o poder de processamento necessários para lidar com os dados gerados pelo gêmeo digital. A computação em nuvem oferece diferentes serviços de computação, incluindo servidores, armazenamento, bancos de dados, redes, software, análise e inteligência, pela Internet. Ela oferece uma abordagem conveniente e sob demanda para acessar recursos de computação compartilhados. Ela permite a distribuição geográfica ao transferir as responsabilidades de processamento e armazenamento de dados para data centers remotos, aumentando a redundância e a confiabilidade do sistema. Embora a computação em nuvem tenha muitos benefícios, como a economia de custos em comparação com uma infraestrutura tradicional local e a capacidade de dimensionar recursos de acordo com a demanda, ela também tem desvantagens. Essas desvantagens incluem possíveis problemas de segurança com a privacidade dos dados, dependência do acesso à Internet, maior complexidade dos aplicativos e possíveis custos adicionais ocultos se o aplicativo precisar ser otimizado corretamente ou se o modelo de preços precisar ser adequadamente compreendido. Além disso, o desempenho da computação em nuvem é estritamente afetado pela latência de rede adicional inserida no sistema para compensar a computação em seus data centers remotos. Aplicações em tempo real com requisitos rigorosos de tempo vinculados à lógica do sistema são exemplos em que o uso da nuvem pode não ser viável, ou pelo menos parte da aplicação deve ser dividida entre o que pode ser implantado na nuvem e o que deve ser executado localmente. A computação de borda pode ajudar a minimizar esses problemas, aproximando os dados de processamento das fontes de dados, em vez de em uma nuvem centralizada, reduzindo o tráfego e a latência e melhorando significativamente os tempos de resposta. Os gêmeos digitais são um exemplo de aplicação que depende muito da integração de vários componentes do sistema e tem requisitos rigorosos de tempo real. O uso de uma infraestrutura baseada em nuvem permite a fácil integração entre suas partes e a acessibilidade dos usuários finais.

TRABALHOS RELACIONADOS: Os serviços de nuvem de gêmeos digitais referem-se a plataformas baseadas em nuvem que permitem a criação, a implantação e o gerenciamento de gêmeos digitais. Os serviços em nuvem oferecem uma série de recursos e funcionalidades, inclusive ingestão, armazenamento, análise e visualização de dados. Alguns serviços de nuvem de gêmeos digitais populares são discutidos nesta seção. A solução de pesquisa proposta não tem como objetivo rivalizar com as ferramentas estabelecidas do setor. Em vez disso, ela serve como uma plataforma útil para estudar e analisar o estado atual do segmento. Várias soluções em nuvem para gêmeos digitais ou para a Internet das Coisas Industrial estão sendo oferecidas atualmente no mercado como

plataformas, como o Microsoft Azure Digital Twins, o AWS IoT TwinMaker e o Siemens Insight Hub, que permitem a criação e o gerenciamento de modelos de gêmeos digitais. O uso de soluções dos principais provedores de nuvem permite a comunicação com outras ferramentas de nuvem oferecidas por eles, oferecendo a possibilidade de conexão com outros serviços de nuvem fornecidos por eles. No meio acadêmico, várias arquiteturas potenciais de gêmeos digitais foram discutidas, com diferentes níveis de integração, tipos de ferramentas de simulação e visualização, com diferentes casos de uso e escala de implementação. Para possibilitar um serviço de comunicação confiável na borda, outras soluções abordaram o uso do Apache Kafka, incluindo a emulação de poços de petróleo, infraestrutura e cenários veiculares.

PROPOSTA: Evoluindo a partir dos trabalhos anteriores, há vários obstáculos para a implementação de gêmeos digitais de uma forma que seja escalável e sustentável. A latência é um problema significativo na comunicação da Internet das Coisas, principalmente em ambientes industriais. Com o aumento da adoção e da implementação, a importância dos serviços de baixa latência se tornará ainda mais evidente. O crescimento da comunicação em larga escala e o aumento da demanda podem limitar a inovação tecnológica relacionada aos gêmeos digitais. Portanto, é fundamental considerar fatores como a escalabilidade do sistema ao implementar os gêmeos digitais. Para superar o problema da perda de mensagens durante a comunicação, alguns *brokers* MQTT oferecem a opção de trabalhar em modo de *cluster*, em que vários brokers são conectados e trabalham juntos para obter melhor desempenho, disponibilidade e balanceamento de carga, o que garante um sistema robusto e tolerante a falhas com alto poder de processamento. No entanto, os problemas de latência persistem mesmo depois que o cluster recebe os dados. Ao lidar com grandes quantidades de dados, é necessária uma infraestrutura robusta para armazenar e processar dados em tempo real. É proposta uma abordagem que consiste nos clientes e na instância digital, que geram e recebem dados, respectivamente, comunicando-se por meio do EMQX, um *broker* MQTT capaz de suportar grandes quantidades de tráfego enquanto trabalha em um modo de *cluster*, oferecendo balanceamento de carga ao sistema. O Apache Kafka também é usado, conectando-se ao *broker* MQTT, para lidar com grandes quantidades de dados, fornecendo redundância por meio da replicação de dados e resistência a falhas na comunicação, armazenando e processando dados em tempo de execução.

EXPERIMENTOS: Os experimentos empregaram uma configuração de rede usando um cluster Kubernetes, diferentemente das abordagens de simulação anteriores que uti-

lizavam apenas máquinas virtuais, para obter resultados mais autênticos e precisos. Os experimentos mostraram possíveis problemas nas execuções do cenário causados pelo fato de os corretores precisarem de mais recursos do que os dispositivos de borda com especificações baixas poderiam fornecer. Eles discutiram a viabilidade do uso de sistemas de processamento de eventos e de vários *brokers* com dispositivos de borda, considerando os recursos dos sistemas e sugerindo possíveis soluções. O experimento 1 apresentou um panorama de problemas causados pelo uso de um único broker MQTT, que causa tolerância zero em caso de falha do *broker*, perdendo todas as mensagens enfileiradas. O EMQX pode atenuar esse problema usando mensagens retidas, que armazenam a mensagem mais recente de um tópico. No entanto, isso resolve apenas alguns dos problemas e força a necessidade de replicação de dados por meio de vários nodos de *brokers*. O experimento 2 vai além disso, agrupando o *broker* em vários nodos conectados. No entanto, a implementação de *cluster* existente realiza uma conexão de malha completa, com todos os nodos armazenando os tópicos inscritos de cada nodo e encaminhando as mensagens recebidas para esse nodo, se necessário. O cenário deste experimento mostra um exemplo do que acontece com a distribuição aleatória de onde os clientes publicarão e onde se inscreverão. Ele apresenta uma longa cauda de tempo extra de apenas um *broker* enviando mensagens para outro *broker*, acrescentando uma hora extra até que todas as mensagens sejam enviadas para os assinantes. As possíveis correções podem ser vinculadas ao protocolo, forçando a reconexão com o agente direto, seja do editor ou do assinante, reduzindo a retransmissão do agente que causa tráfego adicional e afetando os outros tópicos que estão sendo processados nos mesmos nós. As versões mais recentes do EMQX agora apresentam uma nova topologia de *broker*, com apenas nós centrais armazenando os tópicos do *broker* em uma malha completa com outros nós centrais. Em contrapartida, os nodos replicantes se conectam a um nodo central e aos clientes, reduzindo as retransmissões necessárias no *cluster*. O experimento 3 concentrou-se no desempenho do *cluster* de nodos do Apache Kafka, apresentando o novo protocolo de quorum usado pelo Kafka para autogerenciar metadados em vez de depender da dependência externa do ZooKeeper. Os cenários apresentados mostram problemas com o alto uso de memória e a distribuição de computação entre os nós e argumentam que, apesar dos vários benefícios da comunicação, adicionar um aplicativo pesado como o Kafka à arquitetura pode nem sempre ser benéfico para o usuário final. Realizar uma distribuição manual de tópicos, sabendo onde os tópicos estarão e fornecer um link direto para eles, em vez de depender de retransmissões dentro do *cluster*, alivia os gargalos na memória e na CPU, oferecendo suporte a uma

maior taxa de mensagens, mas isso requer intervenção manual no sistema.

CONCLUSÃO: Este trabalho apresentou um estudo sobre gêmeos digitais habilitados para computação em nuvem e de borda, sistemas de réplicas digitais capazes de compreender e expandir as funcionalidades de suas contrapartes reais, explorando os desafios e as oportunidades de implantação e comunicação com um sistema de gêmeos digitais em um ambiente distribuído. Este trabalho teve como objetivo estudar como os sistemas de gêmeos digitais podem ser ampliados usando a computação em nuvem e de borda, testando as necessidades de comunicação em tempo real de um sistema de gêmeos digitais implantado em um cluster de nuvem e de borda com vários componentes. Este trabalho foi desenvolvido com base em pesquisas anteriores na área, e foram realizados experimentos com MQTT e Apache Kafka para verificar possíveis problemas para dispositivos de nuvem e de borda. Os experimentos também discutiram as compensações e limitações do uso do Kafka para gêmeos digitais, especificamente nesses cenários de baixa especificação. Eles sugeriram alternativas e melhorias para trabalhos futuros, considerando as tecnologias usadas. A análise da arquitetura do sistema é fundamental para entender os requisitos reais para executar um sistema como esse. Dependendo do grau de importância da execução, o sistema precisará operar independentemente de uma nuvem central sem causar uma falha nas operações, e a adição de uma camada extra de computação a um aplicativo com muitos dados pode sobrecarregar o hardware acessível pelo sistema de gêmeos digitais, que pode não ter acesso a data centers externos para descarregar a computação. A configuração do Kafka também é complexa, principalmente para usuários não especializados, e requer manutenção para lidar com o particionamento correto de tópicos, aquisição de dados e fatores de replicação. É fundamental compreender os objetivos e as despesas para obter resultados positivos. O custo, que pode ser medido pela infraestrutura e pela complexidade operacional, deve ser gerenciado individualmente. Usando muitas tecnologias, conforme mencionado, para cenários de gêmeos digitais mais complexos, pode-se questionar o valor da otimização de dados em relação ao tempo gasto na configuração. É um desafio mapear a distribuição de todos os tópicos com um conjunto de dados crescente. O rebalanceamento do cluster é outro problema. O tempo perdido na fila de mensagens desperdiça recursos que poderiam ser usados para obter mais dados ou aprimorar o modelo. Um dos principais problemas da modelagem de gêmeos digitais é a ingestão de dados: como obter dados, como aprimorar o modelo de simulação e com que rapidez podemos fazer isso para que o gêmeo digital possa oferecer benefícios. É fundamental entender as metas e os custos para obter bons resultados. O custo pode ser medido

tanto pela infraestrutura quanto pela complexidade operacional e precisa ser tratado caso a caso. A viabilidade e os benefícios do uso da computação em nuvem e de borda para gêmeos digitais são um grande tópico no meio acadêmico, bem como a necessidade de mais pesquisas e otimização para atender aos requisitos de tempo real e aos problemas de gerenciamento de dados desses sistemas, permitindo seu uso no contexto da computação de borda.

PALAVRAS-CHAVE: Gêmeos Digitais. MQTT. Apache Kafka. Computação em nuvem. Computação de borda.

AGRADECIMENTOS: Projeto PeTWIN - Gêmeos Digitais para Otimização e Gerenciamento de Produção [1], liderado pela Professora Mara Abel e o Professor João Cesar Netto, projeto tem vínculo direto de motivação com o trabalho desenvolvido. O projeto é uma parceria entre o grupo Sistemas de Computação para E&P de Petróleo do INF-UFRGS e o Sirius Lab da Universidade de Oslo, viabilizada por meio da Financiadora de Estudos e Projetos (FINEP) e do Conselho de Pesquisa da Noruega (RCN), cofinanciada pelas empresas Petrobras, Shell, Total Energies, CNPC e CNOOC, que compõem o consórcio Libra no Brasil, e a Shell e Equinor norueguesas.

[1] Mais informações podem ser encontradas na página inicial do projeto: `https://petwin.org`

APPENDIX B — A STUDY ON CLOUD AND EDGE COMPUTING FOR THE IMPLEMENTATION OF DIGITAL TWINS IN THE OIL & GAS INDUSTRIES

The following appended document is the preprint version of the paper, with its final version revised, approved and published by Computers & Industrial Engineering, available at `https://doi.org/10.1016/j.cie.2023.109363`. This work was supported in part by the PETWIN project (financed by FINEP and LIBRA consortium), by the Brazilian National Council for Scientific and Technological Development (CNPq) and in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

The paper proposes a systematic research to guide the adoption of infrastructures based on Cloud and Edge Computing for the implantation of Digital Twins in the context of applications for the Oil and Gas industry, considering the relations between Cloud/Edge, Digital Twins, and the O&G industry, individually breaking down 1535 results into a final selection of 56 unique papers, covering the three focus points of the study and citing the different applications, approaches and areas inside the industry using or researching Digital Twins.

A Study on Cloud and Edge Computing for the Implementation of Digital Twins in the Oil & Gas Industries

Francisco Paiva Knebel[a,*], Rafael Trevisan[a], Givanildo Santana do Nascimento[a,b], Mara Abel[a], Juliano Araujo Wickboldt[a]

[a]*Institute of Informatics, Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil*
[b]*Petróleo Brasileiro S.A. (Petrobras), Rio de Janeiro, Brazil*

Abstract

In the Oil and Gas industry, minor accidents can negatively affect people, the environment, and the enterprise on a grand scale. For this type of businesses, it is crucial to have adequate monitoring and maintenance operational routines. The concept of digital twins has been widely discussed as an industrial solution capable of monitoring objects in real-time, predicting their state, integrity, safety conditions, and providing feedback to users. This analysis can also be used to perform predictive maintenance and estimate efficiency levels. This paper follows a systematic approach and inspects the current state of digital twin adoption by the O&G industry, along with cloud and edge utilization in both scenarios. The survey was conducted by filtering and selecting articles published in three different queries regarding digital twins in the O&G industry, cloud and edge usage in digital twins, and cloud and edge usage in the O&G industry. The selected articles were classified and reviewed to present an overview of the current literature regarding digital twins in the oil and gas industry and evaluate cloud-based solutions for digital twins in this context. This review found that cloud and edge adoption in the O&G was late compared to other industries, mainly because of security and data privacy concerns. Digital twin adoption across all industries is still in its infancy, as most of the current works are theoretical or present partial implementations, which often follow different digital twin definitions. Cloud and edge are critical enabling factors for digital twin implementation. Their usage grants access to increased storage and computational capabilities, along with decoupling the necessity of relying on a robust local IT infrastructure.

Keywords: Digital Twin, Cloud Computing, Edge Computing, Oil and Gas

1. Introduction

Until recently, in Oil and Gas (O&G), definitions for Digital Twins were still scarce and non-technical [1], only being used as a commercial term, with no definition of an all-purpose Digital Twin, which could integrate solutions from multiple vendors, from many data sources, and simulation models. For most industries, availability and monetization of data were the primary motivator for digitalization [2], but O&G industry leaders saw no advantage in changing their business model. Besides the fact that O&G industry required highly critical and complex operations, oil profitability rates were high, making O&G companies late adopters of some digital technologies. There were no proven use cases with apparent benefits against known methods that already worked.

For Irving [2], in O&G industry, data science concepts were introduced but motivated by hype, distracting the focus that these technologies must be built with a solid foundation. The O&G industry has data generated from decades of use from physical equipment and assets, with little standardization between manufacturers, which provides no easy way of accessing

*Corresponding author
Email addresses: francisco.knebel@inf.ufrgs.br (Francisco Paiva Knebel), rafael.trevisan@inf.ufrgs.br (Rafael Trevisan), gsnascimento@petrobras.com.br (Givanildo Santana do Nascimento), marabel@inf.ufrgs.br (Mara Abel), jwickboldt@inf.ufrgs.br (Juliano Araujo Wickboldt)

and computing this data, which could be resolved with cloud storage and the usage of open software platforms, reversing the use of proprietary technology and allowing for partners, suppliers and experts to collaborate effectively [3].

Recent works aim to change this scenario, with industry leaders pushing improvements into creating open and interoperable systems, opening the industry into a new era of digitization through the use of Digital Twins (DT) and open cloud-based data platforms for standardization [4]. Digital Twins are composed of at least five dimensions [5]:

- Physical Entity: an object which, through sensors, provides information on its behavior, states, and characteristics. This entity can be abstract, representing a non-concrete entity of the real world.
- Virtual Models: for each physical object, there exists a mirrored digital model in the virtual world through its entire lifecycle. This mirrored model can model the appearance of the object or its behavior but is critical to provide deep insights into the physical entity via simulations.
- Data: in abundant numbers, they provide valuable information for modeling, simulation, optimization, and predictions. Historical and live real-time data are used to generate decisions for the digital twin.
- Connections: essential to enable all elements from the digital twin because both the real and virtual counterparts interact with other entities through both physical and digital spaces.
- Services: encapsulation of functionalities provided by the digital twin. The users only provide inputs and receive results of the functions executed.

Cloud computing is essential in implementing digital twins, augmented with on-demand capacity of external cloud servers, ensuring the scalability of communications, storage, and computation [6]. Cloud-native software architectures also promote the independence between the services layer and the underlying databases through API-based data access. Some DT applications can provide functionalities relying on technology agnostic API-managed microservices distributed over the cloud, and as presented, there are several different implementation options for DTs, with requirements being largely dependent on their intended use case, but encouraging discussion about the specific features being implemented in them [7]. This paper proposes a systematic research to guide the adoption of infrastructures based on Cloud and Edge Computing for the implantation of Digital Twins in the context of applications for the O&G industry. Three pieces of research were made, tying up the relations between Cloud/Edge, Digital Twins, and the O&G industry. This research broke down the 1535 results into a final selection of 56 different papers, covering the three focus points of the study.

The paper is structures as follows: Section 2 *Methodology* presents the strategy used for this research, while Section 3 *Search Results Metadata* complements it by discussing information about the search results and its final selection of papers. Section 4 *Digital Twin and Cloud Fundamentals* provides a brief summary on important aspects of Cloud Computing and Digital Twins, for readers not familiar with the subjects. Section 5 *How Cloud and Edge Computing enabled Digital Twins in other industries* starts the discussion in how cloud-powered digital twins could improve several different fields and some implementations in those areas. Section 6 *Cloud and Edge in Oil & Gas* discusses how was the early adoption of Cloud Computing for O&G, the transition towards Edge Computing, and how they were implemented for the industry. Section 7 *Digital Twin for Oil & Gas* closes the loop by connecting both Digital Twin and O&G, citing different applications, approaches and areas inside the industry using or researching Digital Twins.

2. Methodology

As a systematic research, strict rules were set for selecting the final body of work presented in this paper. First, Google Scholar was used as the primary search tool, returning the list of articles for each query. To achieve the goals of our research,

three different research queries were defined so that we could acquire enough information from all parties involved. The following queries were used:

- Search 1: allintitle: "digital" ("twin" OR "twins") ("cloud" OR "edge")
- Search 2: intitle:digital intitle:twins "oil*gas" OR intitle:digital intitle:twin "oil*gas"
- Search 3: intitle:cloud "oil*gas" OR intitle:edge "oil*gas"

Search 1 results include papers about developing digital twins using cloud or edge computing without limitation of scope or industry. In contrast, Search 2, on the other hand, specifically queries for works done by the oil & gas industry. Furthermore, completing the circle, Search 3 results in any research made by the oil & gas industry that benefited from the usage of cloud or edge computing. Figure 1 illustrates these relations into the separate searches. After obtaining the list of articles and their metadata, a classification of the papers to be included in the final selection was made.

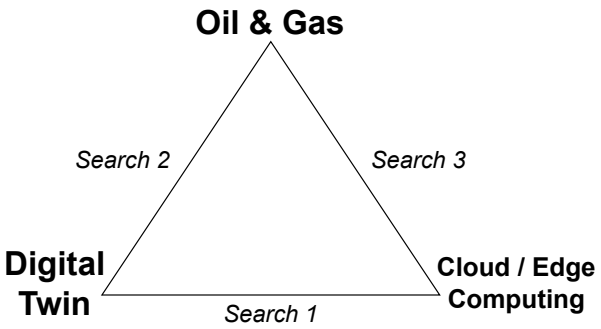


Figure 1: Methodology search queries.

2.1. Exclusion and Selection Criteria

Three selection steps were done to each of the lists returned by the search queries to limit the number of papers that would be selected and remove false positives in our queries. The first exclusion criteria, E1, filtered the initial article list and removed papers that did not have relevance to the subjects expected from each query. It also excluded entries considered improper publications, like magazine articles that present no new research, removal of duplicated entries, and articles not available in English, for review. Full access to the article's text was obtained through various means, including subscriptions to the particular sections of digital libraries, Google Scholar, and contacting authors directly via Academia.edu and ResearchGate networks. The second exclusion criteria, E2, filtered the remaining articles for their publication locations to filter unknown and less reputable publications to prioritize peer-reviewed works with high impact and research visibility. The final selection filters the list of articles again via a manual analysis of the remaining body of work, selecting the most relevant papers on the list. Being a keyword-based search of Google Scholar, results have no guarantee of presenting actual implementations or significant research, only referencing the search keywords at least once in the article, which requires a manual selection of the remaining articles.

2.2. Search Results

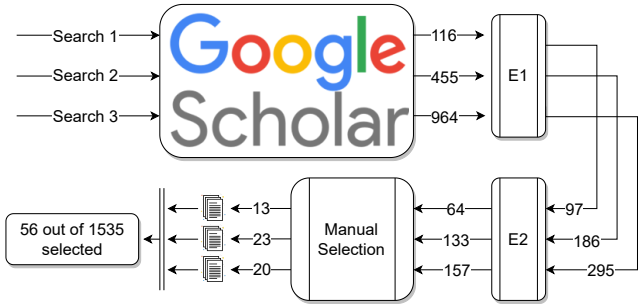


Figure 2: Methodology steps

The final list of papers was obtained by following the search methodology and using the defined exclusion and selection criteria. Figure 2 provides a summary of the flow of steps to obtain the final list. Search 1 returned 116 results, with 19 articles excluded in E1 and 33 in exclusion criteria E2. Out of the remaining 64 articles, 13 were selected. Search 2 returned 455 results, with 269 articles excluded in exclusion criteria E1 and 53 in E2. Out of the remaining 133 articles, 23 were selected. Search 3 returned 964 results, with 669 articles being excluded in exclusion criteria E1 (primarily due to false positives, since "cloud" is a keyword highly used in seismic research but does

not refer to "cloud computing") and 138 in exclusion criteria E2. Out of the remaining 157 articles, 20 were selected. The three queries returned 1535 works, with only 56 included in the final list of papers. This final list will be used in explaining the main sections of this paper: Section 5 *How Cloud and Edge Computing enabled Digital Twins in other industries*, Section 6 *Cloud and Edge in Oil & Gas* and Section 7 *Digital Twin for Oil & Gas*.

3. Search Results Metadata

Being composed of three different search queries, the research contained many publications. Due to its key subjects being cloud/edge (publication growth spur in the last decade) and Digital Twins (growth spur since 2016) being somewhat recent, one can filter the publication corpus by only research publicated in this period. Fig. 3 presents the publication year of the entire corpus. Since the exclusion criteria E1 removed many false positive keywords, we can see that older publications are excluded and the increasing trend of publications in the area in recent years.

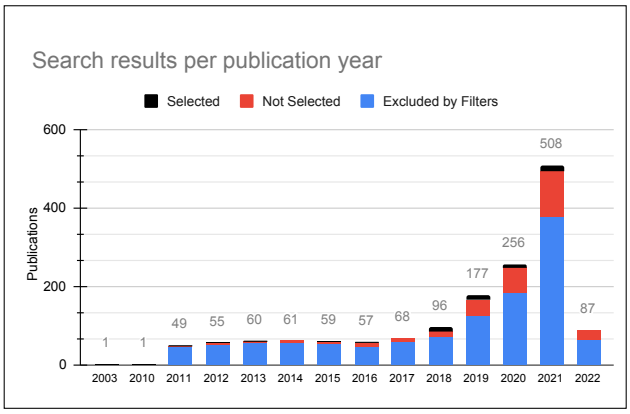


Figure 3: Search results by publication year.

3.1. Selected Publications

From the initial 1535 publication results, only 56 were selected after passing through the exclusion criteria and manual selection. The selected publications share some details on the studies being made on this subject: for instance, Fig. 4 presents the selected publications by publication year, where the limited

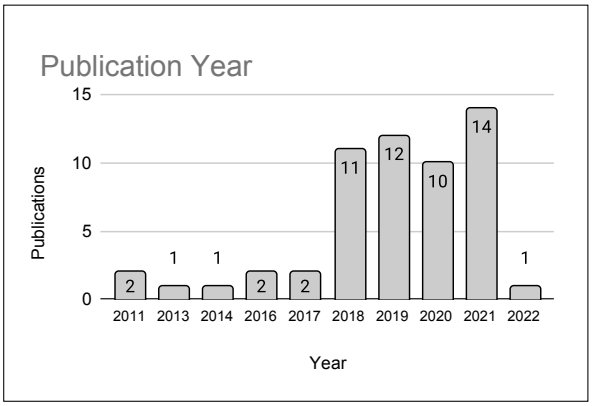


Figure 4: Selected publications by publication year.

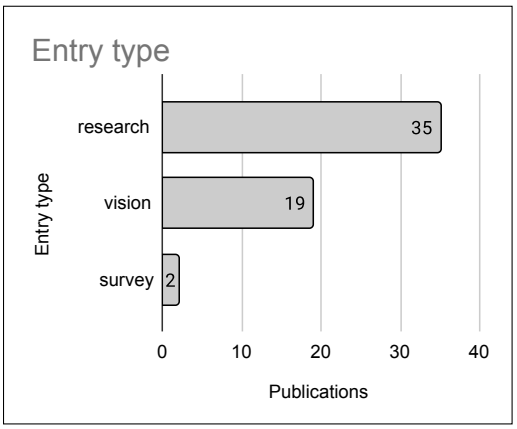


Figure 5: Selected publications entry types.

time period of publications is evident, from the oldest paper selected being published at the start of the last decade and a large amount of research being made in more recent years. Fig. 5 classifies the publications in three different scenarios: research (scientific studies, presenting a problem and a proposed solution by the authors), vision (normally more conceptual papers, not focusing on solutions but presenting possibilities of how the paper subject can be used) and survey (compilation of papers in the area of study, trying to explain the subject of study by combining research made by other authors).

For the selected papers, the distribution of citations is present in Fig. 6. The manual selection of the final list of publications did not focus on the number of citations, as seen by a fairly balanced distribution. The publications with zero citations are recent works considered to be highly relevant and present good discussions on the topic of research. The distribution of affil-

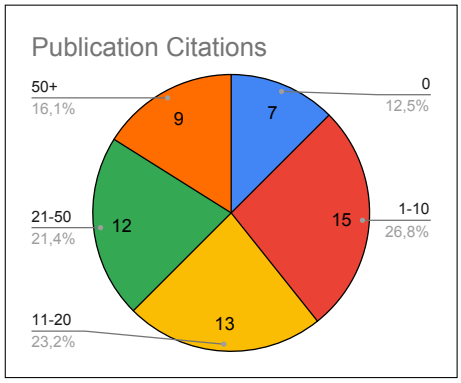


Figure 6: Publication citations.

iation countries, present in Fig. 7, also has a reasonably even distribution but displays where most institutions investing in O&G research are, being evenly split from academic and business backgrounds.

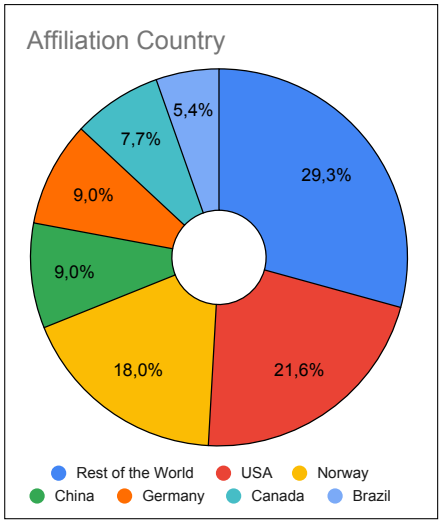


Figure 7: Affiliation country.

4. Digital Twin and Cloud Fundamentals

The Digital Twin is not a relatively new concept. However, its enabling technologies, such as low-cost sensors for the Internet of Things, efficient and intelligent Deep Learning, and ubiquitous Cloud computing, make fully digital simulations of real-life objects a possibility and an approachable reality.

The growth in the usage of cloud computing brought several advantages in building a diversity of different applications at a reduced cost. In contrast, software providers only need to focus on building their applications instead of distributing them to their customers.

4.1. Cloud Deployment Models

Cloud computing providers can make resources available in the cloud through several different forms. One of the standard models is a Public model, where the service provider makes its resources (applications, infrastructure, data storage) freely accessible to users over the Internet. A public cloud model offers limited control into the underlying infrastructure, often not having an option in where the computing is hosted. On the other hand, a Private cloud model is built using the client's resources, creating an "internal cloud", hosting limited access to services behind a firewall. This model is generally costlier due to the need to buy, build and manage the infrastructure. A Hybrid model is the combination of both Public and Private, when the customer wishes different levels of security between applications, using public resources from some providers while also having its internal services [8].

The usage of the Public cloud was questioned in regards to security concerns, in the earlier days of Cloud computing, due to its multitenant model, where different customers share the same hardware, leading to the misconception of the Private model being more secure due to allowing a greater level of control by the customer. Despite this, due to being much larger targets than Private clouds, Public clouds also attract more attention to guaranteeing security by their providers (Google, Amazon, Microsoft, to name a few influential organizations providing public cloud services) and can more easily expand, research and obtain the latest security equipment to support their services. Private cloud apparent higher security due to more restricted access ignores the lack of security when the internal network is compromised, where breaches could be unknowingly created by staff using older out-of-date hardware.

Penetration testing in private clouds is naturally less thorough due to public clouds being large targets and constantly integrating new security measures. Meanwhile, a privately owned infrastructure may have a series of expertly made security tests,

but that gives no guarantee that these provide security from new methods of cyberattacks. In the end, implementation and deployment of a cloud-based application require the decision by its stakeholder on how they plan to control, expand and manage it, either through using general public services provided by large companies, which may not be as customizable and built for their specific use case but that can easily be expanded with more resources on demand, or acquiring the services of smaller providers to help in building an internal cloud, resulting both in higher control, cost and maintenance required. For businesses expanding into the cloud, the question is: should they focus on tailoring its deployment or the application itself? The focus should be on dedicating time and resources to the main application instead of an area where the customer might have no experience.

4.2. Cloud Service Models

The cloud provider, being the entity in control of the underlying hardware, can provide its cloud computing services on different scales, depending on the product and intended use by its customers [9]. The most widely used services models are:

- **Infrastructure-as-a-Service** (IaaS): the customer hires specific hardware on demand from the cloud provider. Depending on the supplier, this could include hiring servers, network routers, and racks. This is typically done through data centers with cloud-hosted virtualized servers in a pay-per-usage manner. In this model, customers are responsible for the configuration of this infrastructure in order to run its applications.
- **Platform-as-a-Service** (PaaS): provides the computing resources for consumers to develop their applications with their data, but with less control on the specific infrastructure used. In this model, the customer only needs to focus on developing the application. At the same time, the responsibility of configuration and deployment of infrastructure is left to the cloud provider.
- **Software-as-a-Service** (SaaS): in this model, the cloud provider gives access to an already built software to the customer, a ready-to-use cloud-hosted application that he can use through the cloud without worrying about configuration, deployment, and maintenance. This is the most widely used model by internet users via email clients, social media websites, or online versions of native software.
- **Serverless/Lambda/Function-as-a-Service** (FaaS): instead of constantly running processes on a server, FaaS has an event-based mechanism, triggering the execution of just a piece of code on demand. The customer does not have to consider the server executing this code: the cloud provider is responsible for the execution and scale as necessary.

PaaS applications are similar to FaaS because they "hide" servers from the developers. However, they always have at least one server constantly running, waiting for requests, and scalability requires allocating additional servers, which the customer will be directly charged for, and configuring the application to take advantage of the additional hardware. On the other side, FaaS requires no intervention by the customer for scalability, where the cloud provider handles allocating computation time to execute each function as demand increases. Since customers only pay for the execution of the function (and not the idle time between executions), this will generally lead to lower costs and higher scalability, at the cost of latency for the initial function execution [10].

There is no ideal service. These different models allow for customization of the level of service being hired from the cloud provider, allowing the customer to choose depending on the functionalities required and the expertise of its staff.

4.3. Cloud and Edge Computing

Cloud computing allows on-demand access to a pool of shared computing resources. Data processing and storage are offloaded to remote, geographically distributed data centers, generating system redundancy and reliability. Edge computing moves

processing nearest to where it is needed, allowing computation closer to the source, reducing cloud traffic and service latency, and improving response times for the end users [11].

4.4. Digital Twin

Digital Twins are not just a digital model or interface of a physical object. The concept is of a twin receiving input from various sensors from their real-world counterpart. With the acquired data, the Digital Twin can simulate the physical object it mirrors in real-time while feeding the real system insights into its performance and potential problems. The purpose of a Digital Twin is not to act as a prototype of physical systems, helpful in aiding in the development process, but to provide another level of modeling, where digital and physical coexists in a continuous feedback loop, with the real system providing data and the digital system providing suggestions to the real counterpart, improving its processes autonomously. Both the real-time and simulation aspects must be present to make an application a digital twin. When only the simulation aspects are present, the application is a *digital model*. When real-time visualization is the only concern, the application is called a *digital shadow* [12].

The recent expansion of low-cost sensor devices due to the Internet of Things is one of the main factors pushing towards the development of Digital Twins, being an entirely digital and easy-to-implement solution, without a physical duplication of the real system but only the definition of the system components and feeding their data through it. Implementing a cloud-based Digital Twins allows for integration between its parts and ease of access for the end-users, accelerating product development and manufacturing. Components from the software platform can be expanded, and scaled on-demand [13].

5. How Cloud and Edge Computing enabled Digital Twins in other industries

Among the surveyed articles, three of them presented no implementation but instead focused on the systems architecture or how digital twin technology could improve their respective fields.

Alam and El Saddik [6] describes a digital twin framework in which every device is represented as a cloud-based digital twin. This architecture is designed in a hierarchical manner, where a higher-level digital twin is composed of several smaller digital twins in a master/slave relation. In this framework, the high communicability of traditional cyber-physical systems is augmented with the higher computation and capacity storage of cloud servers. This ensures the scalability of communications, storage, and computation capabilities. A digital twin can then analyze the full system and recommend actions to its physical counterpart to improve its performance.

Dong et al. [24] and Lu et al. [25] both focus on edge based architectures. The former writes about increasing the energy efficiency in 5G services in MEC systems by using deep neural networks. A digital twin that can replicate the MEC could improve resource allocation and offloading probabilities. The latter also explores the usage of digital twins to replicate a network but focuses on using the associated edge for machine learning, using federated learning. In this context, twinning a wireless network allows for real-time data processing to be migrated between edge devices. This, along with blockchain-based authentication, can improve the efficiency, reliability, and security of federated learning models.

The remaining articles have been categorized in table 1, ordered by publication year. In this table, the reviewed articles were categorized in regard to the Industry, Application Scenario, and the usage of Cloud, Edge, and Data Transport. The Industry column shows what kind of industry the reviewed article is geared towards. The Application Scenario column contains the niche the designed application covers. The Cloud Usage and Edge Usage columns show what kind of tasks were offloaded to cloud servers or edge devices, and the Data Transport column indicates what protocols and data formats were used in these communications. For this table, the corresponding node was left blank when the author didn't comment on that particular aspect.

In the Industry column, some of the most recent studies focus on Smart Vehicles, such as Zhang et al. [21] and Lu et al.

	Industry	Application Scenario	Cloud Usage	Edge Usage	Data Transport
[14]	Smart Manufacturing	Manufacture Execution Systems	Storage, UI and computing		MTConnect, HTTP and XML
[15]	Smart Manufacturing	Connect multiple factories	Storage, UI and computing		MTConnect, HTTP and XML
[16]	Healthcare	Healthcare service for the elderly	Storage, analytics, machine learning and UI		Bluetooth, LoWPAN or ZigBee
[17]	Healthcare	Detect cases of an IHD or strokes	Storage and analytics	Analytics	Bluetooth or SMS
[18]	Electrical Engineering	Build a Digital Twin for battery systems	Storage and analytics		MQTT
[19]	Smart Manufacturing	Implement robot control as a service	Storage and UI		
[20]	Smart Manufacturing	Scalable communication	Storage and computing	Storage and computing	MQTT or CoAP
[21]	Vehicular	Improve the learning efficiency of multiple agents		Machine Learning (DRL)	
[22]	Vehicular	Mitigate unreliable and long-distance communication	Storage and analytics and Machine Learning (DRL)	Computing	
[23]	Vehicular, Smart Manufacturing	Data security	Storage, UI and machine learning	Machine learning	HTTP, MQTT, DDS or CoAP

Table 1: Selected cloud and edge computing proposals for Digital Twin implementations.

[22], whose research was driven by the rise of edge computing power and vehicular networks. Liu et al. [16] and Martinez-Velazquez et al. [17] explore the usage of digital twins in the healthcare industry for the prevention of diseases or to provide quality real-time healthcare to the elderly. Most of the other surveyed articles in this column fall under the Smart Manufacturing category with a focus on improving productivity in industrial environments.

The Cloud usage and Edge usage columns show to what end these technologies were used in the reviewed articles. The usage of the cloud as storage was present in every reviewed paper. Most of the earlier articles also used the cloud for UI, with cloud rendered 3D models as suggested by Xu et al. [19], or through the usage of GUI's accessible through web apps as done by Coronado et al. [14]. In older studies, heavier data processing and analytics were relegated to the cloud, as it has better access to resources, but, with the increased capabilities of edge devices, Cathey et al. [23], Lu et al. [22] and Zhang et al. [21] implement Edge computing with the usage of edge association techniques, for heavy data analysis or machine learning.

In the Data Transport column, we gathered information about protocols and data formats used in the experimentation. In earlier works, like Coronado et al. [14] and Hu et al. [15], the HTTP protocol was regarded as a good option, utilizing MTConnect formatting and the XML data format. The Healthcare twins proposed by Liu et al. [16] and Martinez-Velazquez et al. [17], opt for the Bluetooth protocol. More recent articles, with focus on industrial usage of digital twins, such as Li et al. [18], Bellavista et al. [20] and Cathey et al. [23] opt for the usage of more robust protocols such as MQTT or CoAP.

This shift into more digital environments and digital twin adoption pushes industries to rely on a cloud-based architecture for data storage, management, and increased computing power. The smart manufacturing industry can create a more connected and responsive environment to increase safety and efficiency through the cloud and edge usage. The healthcare industry can use IoT to monitor and diagnose patients in real-time, and the vehicular industry can use the edge for task offloading in smart or self-driving vehicles.

6. Cloud and Edge in Oil & Gas

The adoption and use of cloud and edge computing-based solutions by Oil & Gas industries have been a topic under discussion for over a decade. Early investigations, such as those by Feblowitz [26], Beckwith [27], and Perrons and Hems [28], have discussed the main struggles faced and potential benefits of adopting cloud-based technologies to streamline the industry's processes.

The second wave of studies [29, 30, 31] discussed the evolution of the centralized cloud computing model, taking advantage of the distributed edge computing approach to accommo-

date particular applications and scenarios.

Over the last few years, several proposals have been made to employ both cloud and edge computing technologies to solve various problems in the Oil & Gas industries. This body of work is discussed in detail in the following subsections.

6.1. Adoption of Cloud by the Oil & Gas Industry

Considering the work presented in the surveyed body of papers, we selected a series of studies that list important factors towards adopting cloud computing, published in conferences, magazines, and journals relevant to the Oil & Gas industry. One of the most recent papers included in the survey by Lawan et al. [32] covers most of the factors mentioned by previous investigations, aggregating with explanations on the meaning of each factor for the oil and gas industry. The authors map the cloud computing adoption factors using the Technology-Organization-Environment (TOE) framework, which maps the innovation process within enterprises according to Technological, Organizational and Environmental contexts, presented in Table 2 and also including social and political factors that may affect its implementation. The factors presented in Table 2 provide a summarized view of how the oil and gas industry understands the challenges in adopting cloud computing to their industrial demands.

Feblowitz [26] is a significant vision paper that first surveyed Oil & Gas companies regarding their familiarity and possible shift towards cloud computing in the industry in 2011. Despite the popularity of cloud applications, oil and gas companies faced a slow start, with barely any financing for its adoption. This survey highlights the initial struggle that cloud adoption faced, with doubts about support for the existing legacy infrastructure that companies have already built over the years and data security, through reluctance in trusting the safety of data storage outside of the company's control, with concern not only of possible cyberattacks disrupting functionalities, but being open to leaks of trade secrets and sensitive data. These points lead toward the initial adoption of private over the public cloud.

Beckwith [27] mentions the importance of outsourcing for managing big data in the cloud. Due to the large and increasing amount of data being generated, in-house solutions are no longer viable, with its solution being the usage of large, outsourced private clouds, so O&G companies no longer have to worry about technology and data maintenance. Specifically, the author comments on the importance of data storage for seismic surveys, the largest datasets in the industry, and the need for sharing this data across the network with minimum latency for further processing. Due to the complexity and scale of data, repeating surveys is not a viable alternative, demanding reliable storage, both being protected from cyberattacks and outside elements, like protection from heat, water, and electricity outages.

Perrons and Hems [28] identify other additional factors in cloud adoption, such as deployment flexibility, allowing for allocating the required amount of computing and data storage resources according to demand, and implementation speed, due to the short time necessary to bundle and offer cloud services, and the usage of off-the-shelf computing software (or as now more widely known, "apps"), where if they are cloud-hosted, their computation is offloaded to external servers, no longer being dependent on computation by the user's machine. The authors point out the challenges facing the upstream oil and gas industry compared to other industries and the usage of public/hybrid/private clouds.

Al-Mascati and Al-Badi [33] surveyed the adoption of cloud computing in several O&G companies and presented the cloud adoption factors according to the TOE framework. With these factors, the authors surveyed the companies on how they affect the adoption of cloud computing through 14 different hypotheses distributed among the different firms IT managers. Most importantly, the results show that for the majority of received answers, data hosted within cloud computing service providers are considered to be trusted to preserve the security and confidentiality of received data, despite the importance given to data security in other research. Adequate communication service support is a significant factor to consider due to the remote locations of industry exploration sites.

Technological	Organizational	Environmental	Social & Political
Security, Trust, Privacy, Integrity	Technology readiness	Perceived industry pressure	Culture
Reliability, Availability, Accessibility	Top management support	User/technical support	Government support
Compatibility	Workers' attitude	Compliance	
Complexity	Existing IT apps & infrastructure	Internet availability	
Trialability			
Cost			
Relative advantage			

Table 2: Cloud computing adoption factors for the Oil & Gas industry modeled by Lawan et al [32].

6.2. Transition to Edge computing

A centralized cloud computing model presents several benefits, like limitless resource availability and initial investment reduction by outsourcing IT infrastructure acquisition and operations. Nevertheless, recent studies have supported a new trend to redistribute computing workloads back to the edge. Settemsdal and Bishop [30] discuss this matter from a security and privacy standpoint. The authors claim that data privacy is a primary concern for Oil & Gas companies and, therefore, one of the main drivers for using edge analytics. Companies and their supervising governments often place limitations on where data can reside. They want to protect sensitive operational data like well production or capacities. Keeping data at the edge, instead of a centralized cloud, is a means of protecting the data from unauthorized access or leakage. In some cases, Settemsdal and Bishop [30] suggests that on-premise edge analytics may be the only viable option to satisfy data privacy requirements. However, in situations where there are no specific governmental restrictions on data location, it would be prudent to examine which subsets of data are truly sensitive or not to balance between shielding private data and gaining maximum visibility into operations.

Settemsdal and Bishop [30] also discusses the risk of cybersecurity attacks in cloud and edge environments. The authors note that there are numerous cyberattacks attempted daily on industrial facilities. Oil & Gas assets are considered, in many cases, to form a critical infrastructure. Thus, providing a secure environment for deploying the analytics at the edge is of utmost importance. Operators must ensure a secure IoT ecosystem with minimal negative impacts on operational performance. When data analytics procedures can be performed remotely at a central cloud, it makes sense to find the means of securely streaming data into a cloud facility, outsourcing the cybersecurity risks to the cloud provider.

Tedeschi and Sciancalepore [31] raise another concern closely related to a particular characteristic of the Oil & Gas industries. Oil & Gas extraction sites are usually installed in remote locations, and providing high-speed and reliable internet connections is challenging. According to the authors, edge computing can provide significant improvements for Oil & Gas large-scale critical infrastructure deployments by reducing processing time, increasing exploration systems uptime, heightening productivity, and reducing energy consumption. Although the use of edge technology can bring definite improvements, particularly in terms of latency, the usage of these technologies also introduces unforeseen threats such as data leakage and lack of data integrity. To address these issues, the authors suggest that Oil & Gas companies should combine IoT, machine learning, and cloud computing technologies to achieve pervasive and ubiquitous remote facilities management.

Saghir et al. [29] also advocates for using edge technologies as an enabler to unlock potential in upstream Oil & Gas operations. For instance, the authors highlight that machine learning applications can now be embedded directly at well-heads resulting in improved data management, immediate corrective responses, and decreasing losses. Proactive remote management

and well inspection enabled by edge computing minimizes the exposure of human personnel to health hazards and improves operating expenses in upstream operations. The authors also suggest that data can be acquired at the edge using cellular communication using IoT protocols, like Message Queuing Telemetry Transport (MQTT) and Advanced Message Queuing Protocol (AMQP), ensuring secure data transfer over a public network using low bandwidth and consumption. Of course, that is possible in locations where cellular services are an option, referring back to the connectivity concerns raised by Tedeschi and Sciancalepore [31].

Finally, Saghir et al. [29] recognize that the edge is inevitably a resource-constrained environment, particularly in contrast to a cloud environment. The authors mention disk space as a limiting factor at the edge (usually, at best, a few gigabytes per device). In the context of rod pumps, for example, a dynagraph card is generated every few seconds. To provide real-time pump performance indicators, inferring the card shape at every stroke is necessary. Considering the high rate of predictions (a dozen per minute), storing of images and logs, and the resource limitations of an edge gateway, this can quickly become an issue. These issues can be circumvented by carefully managing data files stored at the edge.

6.3. Implementation of Cloud and Edge computing

We have selected and organized some of the major cloud and edge-based proposals of both conceptual and practical approaches, solutions, and tools for the Oil & Gas industry. The result of this selection is presented in Tables 3 and 4. Table 3 organizes the main aspects of cloud and edge computing for each selected study. On the cloud dimension, we captured the Service Model (*e.g.*, IaaS, PaaS, SaaS), the Deployment Model (*e.g.*, public, private, hybrid), and other implementation details in Computing, Storage, and Integration Approaches (*e.g.*, use of event streaming tools, map-reduce). We captured the Connectivity/Transport technologies employed on the edge dimension, considering both communications between field devices (sensors, cameras) and the edge computing devices and from the edge to the cloud, when applicable. We also extracted information on the Platforms used to implement the edge-based applications. It is worth noting that some studies consider only cloud aspects, others only edge, whereas some do both.

Table 4, in turn, summarizes the main aspects of the Oil & Gas application and target industry, as well as the approach taken to develop the proposed solution. In the Target Industry column, we included information on the specific industry sector (i.e., upstream, midstream, or downstream) to which the proposed solution or approach is applicable. In the Application column, we captured the most relevant information on the application context of the proposed solution (e.g., fault detection, plant monitoring, big data loading, processing, visualization) within a specific industry. Some of the proposals are not targeted to a specific application. However, most studies present at least one strong use case from which we can derive the application context. Finally, the Approach column summarizes information on the approaches, mechanisms, or technologies (e.g., machine learning, simulation software) employed to solve the particular problem.

Regarding Deployment Models, at least among the selected studies, it is possible to see a tendency to rely on public clouds rather than private deployments. That does not necessarily mean this tendency would also hold for real production environments. However, this observation supports the idea that the Oil & Gas industries should evolve from private infrastructures to public pay-as-you-go models over time. Since research-oriented proposals tend to be more forward-looking, this phenomenon could be interpreted as a suggestion of where the Oil & Gas industry "wants to be" a few years from the publication dates of the mentioned studies. Interestingly, among the selected studies, there was hardly any mention – not to say actual implementation – of hybrid cloud deployments. For instance, Khodabakhsh et al. [35] suggests using open-source distributed frameworks for big data to manage, process, and analyze sensor data in Oil & Gas applications. The authors propose a private cloud architecture to unify data and analytics, with batch processing for offline data (Cassandra), a serving layer for indexing and vi-

	Cloud			Edge	
	Service Model	**Deployment Model**	**Computing, Storage, Integration Approaches**	**Connectivity/Transport**	**Platform**
[34]	IaaS	Public Cloud (AWS EC2 S3 VPC)			
[35]	PaaS, Serverless, Lambda	Private or Public Cloud	Event Streaming (Apache Kafka/AWS ® - Kinesis), Map Reduce (Apache Spark/AWS - Elastic MapReduce)		
[36]				Pub/Sub MQTT to central cloud	Microsoft ® IoT Edge - Ubuntu Core (IoT), Container-based Virtualization (Docker)
[37]	IaaS	Public Cloud (Azure)			
[38]	PaaS	Private	Event Streaming (Apache Kafka), In-memory computation (Apache Spark), Storage (Cassandra), Visualization (INT Geotoolkit)		
[39]	Serverless, Lambda			Satellite to central cloud and federated edge	EdgeCloudSim simulator
[40]	SaaS, PaaS	Public Cloud (Arundo ®, over any public IaaS))		Delay tolerant, intermittent access, MQTT (to cloud), OPC UA, Modbus (at edge)	Arundo ® Edge Agent
[41]		Public Cloud (Many providers)	Big data management and exchange (OSDU), Data Availability (Diskos), Data Integration, exploration, quality (EarthBank)		
[42]				Wireless cameras is only used only locally no cloud connectivity	
[43]		Public Cloud (Azure)	Storage (PostgresSQL, TimescaleDB on Azure), Data Exchange (WITSML, and public cloud IoT platforms)		In-house proprietary "Edge" platform (NOV ®), Kubernetes service mesh in the cloud
[44]		Private (centralized in a so called "server")		Wireless 4G, 5G, GPRS	
[45]	IaaS	Public Cloud (AWS ® Elastic File System (EFS))			

Table 3: Selected cloud and edge computing proposals in the Oil & Gas industry - Cloud and Edge aspects and technologies used

sualization (Spark), and a speed layer for real-time processing (Kafka). The authors also suggest that these components could be integrated with public cloud-compatible services (e.g., Amazon's Elastic MapReduce, Kinesis, and S3). Although they do not explicitly claim that their architecture supports hybrid cloud deployments, it cloud be interpreted as such.

There is a mix of uses of the leading service models available on the Service Model dimension, favoring lower/mid-level resource abstractions, such as IaaS and PaaS, rather than higher-level ones (i.e., SaaS). A common approach taken by some of the studies is to take some piece of software or software components that companies used to run on desktop computers or a local pool of servers and deploy them partially, or in full over virtualized cloud resources (virtual machines, in general) [34, 37]. One of the first investigations to take this approach was published by Eldred et al. [34]. The authors presented a pilot project based on AWS ® public high-performance cloud computing. They take an industry standard reservoir simulation software (i.e., ECLIPSE ®) that was initially designed to run on a workstation or local server and adapted it to run over virtualized resources on the cloud. The main benefit of doing so are flexibility, accessibility, and cost reduction (pay-per-use). The authors also present how to calculate the break-even point of deploying the system on a public cloud versus an internal infrastructure through an illustrative cost model, discussing the relationship between the hourly cost of infrastructures and resource utilization.

Among the few mentions of SaaS as the adopted Service Model is the proposal of Wen et al. [40]. The paper focuses on using federated learning to mix the centralized cloud and decentralized edge training models. However, reading through the references, the same authors published a previous study

[46] in which they detail the implementation and deployment of the cloud/edge platforms used in their work. The foundations for the platform run over any public IaaS provider (e.g., AWS, GCP, Azure), whereas the authors propose a PaaS framework to develop the software layers required to interact with the distributed machine learning models and associated training data. The solution is intended to bridge the gap between data science teams (SaaS developers) and field operators (SaaS users).

On specific Computing, Storage, and Integration Approaches, data stream processing mechanisms stand out to provide scalable big data services for oil and gas industries, allowing high reliability and high performance on real-time or near real-time services. For example, Yang et al. [38] leverages open source big data frameworks (e.g., Apache Spark and Kafka) and machine learning techniques toward autonomous analysis of data sources. The authors use the proposed solution to efficiently process enormous and complex datasets to provide real-time or near real-time data analytical service, including prescriptive and predictive analytics. Also, on the big data management subject, another everyday use case for cloud-based solutions is the storage and processing of geoscience and seismic-related datasets [41, 45]. Oikonomou et al. [41] approach the challenge by relying on cloud-based big data management and exchanging open solutions, such as the Open Subsurface Data Universe (OSDU), which is an open data platform developed as part of the Open Group global consortium[1]. Guimaraes et al. [45] show significant advantages in converting and storing data from large SEG-Y files into commercial public cloud services, such as the distributed cloud-based Elastic File System (EFS) by Amazon.

Regarding Connectivity and Transport aspects and mechanisms adopted by the selected publications, there is an invariant among every proposal related to edge computing: communication resources are presumably intermittent and unreliable. This is because many Oil & Gas facilities are installed at remote locations with poor quality or expensive connectivity services. Some authors, like Hussain et al. [39] assume there is satellite connectivity from the edge to the central cloud or among a federation of edge facilities, while others like Huan et al. [44] suggest cellular connectivity. For example, Pink et al. [43] proposes an end-to-end solution (edge platform) designed for both the field and office requirements. The system is designed to duplicate capabilities in the cloud so that if cloud connectivity is lost, then "full functionality" could continue on the remote (edge) location. However, the edge is usually assumed to be an environment where computing, storage, and communication resources are scarce (different from the cloud, where they are assumed to be virtually "infinite"), in the particular case of the Oil & Gas industries the edge needs to be robust enough to operate even without the cloud. Gooneratne et al. [42], for instance, proposes a standalone edge deployment with no cloud connectivity whatsoever.

Among the edge platforms employed in each surveyed study, there is a mix of available tools and frameworks, such as Microsoft IoT Edge (based on Ubuntu Core IoT) [36] and proprietary solutions, such as Arundo ® Edge Agent [40] and NOV ® in-house edge platform [43]. Some studies either do not mention specific implementations of edge platforms or do not provide enough details on their implementation. Only one investigation by Hussain et al. [39] relied on a simulator (EdgeCloudSim) for validating their proposal.

Finally, it is essential to note that not all studies in our survey explicitly mention a specific service/deployment model, tools, protocols, and platforms they adopt. In many cases, we had to infer from the discussions or from reading related resources (i.e., software documentation, company websites).

7. Digital Twin for Oil & Gas

Building a Digital Twin application for O&G would provide a digital environment where engineers could experiment and try various scenarios for problem-solving without impacting the platform's productivity. Integrating information provided by the multiple sources involved in production platforms, ranging from physical objects to data generated by specialized

[1]For further information on OSDU and the Open Group visit: `https://osduforum.org/about-us/who-we-are/the-open-group/`

	Application	Approach	Target Industry
[34]	Reservoir simulation	Simulation (ECLIPSE ®)	Upstream (Reservoir)
[35]	Sensor data fault detection	Machine Learning (Least Square Estimation, Auto-Regressive Moving Average, and Kalman Filters)	Downstream (Refinery)
[36]	Rod Pump Failure detection (Dynagraph Card recognition)	Machine Learning (Transfer Learning, Ensembling, Convolutional Neural Network, Siamese Neural Network, Autoencoder Neural Network, Histogram of Oriented Gradients)	Upstream (Artificial Lift systems)
[37]	High-resolution reservoir simulation	Machine Learning (genetic algorithm, particle swarm optimization and ensemble-based optimization), Simulation (INTERSECT ®)	Upstream (Production, Field Development)
[38]	Fiber-optics monitoring, distributed sensing data from distributed sensors (acoustic, temperature, strain, etc), Time-lapse seismic	Machine Learning (Principal Component Analysis, Linear Regression, Least Median Squares, Hidden Markov Models)	Upstream (Production, Reservoir)
[39]	Fault/Disaster Management (Task Allocation)	Heuristic based on Probabilistic	Upstream (Production)
[40]	Failure Detection, Cyberattack Detection, Automated Logistic Management	Machine Learning (Mesh, Federated, and Hierarchical Learning)	Up and Midstream
[41]		Machine learning (TensorFlow ®, EarthNet ®)	Upstream (Exploration)
[42]	Drilling Optimization	Machine Learning (Deep Learning, convolutional neural networks (CNN), single-shot detector (SSD), and you only look once (YOLO) with TensorFlow ®)	Upstream (Drilling)
[43]	Customer defined and 3rd party applications (plug-ins) for data analytics, Insight based on data visualization	Drilling Optimization based on Electronic Drilling Recorder (EDR) data (simulated)	Upstream (Drilling)
[44]	Pipeline monitoring		Midstream (Transportation)
[45]	Seismic data loading (performance/cost analysis)		Upstream (Exploration)

Table 4: Selected cloud and edge computing proposals in the Oil & Gas industry - Applications and Approaches

staff, can enhance management and collaboration, preventing mistakes and rework.

A cloud-based platform would provide limitless access to information between different teams so that the right decisions can be made while missteps are avoided. Asset failures can be predicted, while hidden potential can be discovered, allowing for continuous improvement in Exploration & Production processes while reducing risks.

Wanasinghe et al. [4] and surveyed the O&G industry, considering Digital Twin adoption. The authors found that innovations like IIoT, machine learning, and big data are some of the technologies that are enabling the move of the O&G industry from a traditional business model into a more digital one, but the adoption of digital twin technology is still in its first stages. The industry's high concern with data privacy and security could be one of the factors keeping them from investing in establishing digital twins linking their physical devices in a physical space. Proper implementation of digital twins is not yet found in the industry. Conversely, Cameron et al. [1] surveyed digital twin articles and found that explicit references to digital twins in the oil and gas industry are few and are often high-level, non-technical presentations. The authors believe that semantic backbone (ontologies) to support integration are necessary to standardize semantic models.

While Avanzini and Eriksson [61] argues that the lack of trust limits the value of DT in the O&G industry and that they will function as specified, LaGrange [62] assesses that there is still a great deal of confusion regarding what digital twins are and how they can add value to O&G operations. The former believes that the DNVGL-RP-A204 "Qualification and assurance

	Application	Approach	Target Industry
[47]	Visualizing and interacting with DTs using AR	Automation of data acquisition and interactive visualization (Augmented Reality)	Offshore Oil Platform, Upstream
[48]	Building a well, comparing data from previous drills with data acquired from the current drill	Machine Learning and Simulations	Upstream (Drilling)
[49]	Presents an abstracted DT model for an offshore platform	Machine Learning	Offshore field, Upstream
[50]	Fatigue life prediction of marine structures (offshore platform)	Simulations and 3D Modeling (SIMA)	Offshore, Production
[51]	Automated monitoring of drilling wells	Simulations, mathematical modeling	Upstream (Drilling)
[52]	Digital Twin applied to an offshore field environment	Automation of data loading, transformation, standardization, 3D visualization	Offshore field, upstream & downstream
[53]	Building a well, utilizing cloud-based simulations to monitor the drilling performance	Simulations (DynCap)	Upstream (Drilling)
[54]	Production control and optimization	Machine Learning	Upstream
[55]	DT in drilling, to plan, monitor and operate well construction	Drilling simulation, purpose-specific algorithms for monitoring and diagnosis	Upstream (Drilling)
[56]	Drilling optimization	Automated Machine Learning (AutoML)	Upstream (Drilling)
[57]	Presents a decentralized computational framework for Digital Twins	Machine learning and simulation	Offshore and onshore fields, pipelines
[58]	Digital model of an FPSO, with model construction, simulation and data processing	Simulation (OrcaFlex)	Offshore, FPSO
[59]	Fault detection (Prognostic Machine Learning)	Data Analysis, Machine Learning, Storage (Cloud)	Midstream
[60]	Well control, creating a DT of the drill	Simulations	Upstream (Drilling)

Table 5: Selected case studies with Digital Twin implementations in the Oil & Gas context.

of Digital Twins", the first standardization on developing and operating a digital twin, will increase industry trust in the quality level of the service provided by this technology. The latter elaborates on how digital twin is still used as a synonym for 3D models and how outlining its components is critical in defining what it is.

Laborie et al. [63] describes the creation of a foundational digital twin by aggregating the shadows of several assets. The linkage of several models could provide operators with a comprehensive and real-time view of a complex structure and access to historical data. Akbari [64] also have concerns regarding data visualization. The authors propose a way of visualizing data through an augmented reality where the information retrieved from sensors is mapped into a 3D model and overlayed with the real twin. This way, a user can access data from the digital twin by visualizing the real structure through smart glasses.

For Said et al. [65], as the amount of generated data increases, more automated and complex processes must be done in order for operators to follow them efficiently. The authors propose a Digital twin model to automate well and run real-time diagnostics through simulation and machine learning. In this context, digital twins can be used to both automate systems and shift the focus of drilling operations from efficiency to safety. Abdrakhmanova et al. [66] researched a risk management method called "tree of consequences" and how it can be implemented in a digital twin context. The tree of consequences is a method used to predict malfunctions, taking into consideration safety measures to assess the most critical points of failure.

Combining this method with a digital model provided by simulators such as FLASK and Phast Lite, it is possible to simulate the release of hazardous substances and calculate possible accident scenarios. Mayani et al. [67] also studied the usage of hydraulic and temperature simulations for digital twin fault detection. Using these simulations' results, operators can keep a tight margin on problematic conditions and optimize drilling conditions. This will increase safety in the drilling operations, where problems cause a significant economic loss.

7.1. Case Study Analysis

Table 5 was assembled in the same way as Table 4, but it categorizes the articles from the "Digital Twin / Oil & Gas" query that presented some form of implementation, ordered by publication year. In the Application column, the objective of the application was summarized. The approach column contains which method was chosen to implement the digital twin solution, and the industry column shows which part of the O&G industry research is geared towards.

In the Approach column, Schroeder et al. [47] focuses on data visualization in a digital twin using augmented reality and the ability to interact with it using web services. Tygesen et al. [50] and Brewer et al. [52] also have concerns with data visualization, but opt for the usage of 3D models. The remaining articles focus on machine learning or simulation approaches for digital twin implementation. Using these algorithms, digital models can either predict malfunctions by analysis of historical data or comparing the acquired data with values generated by physical models, and in both cases, it can signal for predictive maintenance when necessary.

Regarding the target industries, Priyanka et al. [59] and Chowdhury et al. [57] are concerned with integrity and safety in oil pipelines. Priyanka's work brings up fault detection in the midstream, using prognostic machine learning techniques, while Chowdhury proposes a framework for oil field deployment encompassing offshore fields, onshore fields, and the oil pipelines. Rangel et al. [58] shows a particular interest in FPSOs using simulations and predicting their behavior to prevent them from receiving major damages. Tygesen et al. [50] cite their concern with offshore platform integrity and compare real-time data with simulations to guarantee the structure's safety. The remaining articles primarily target the upstream industry, focusing on the drilling stage.

The cloud adoption approach is discussed by some of the authors of the surveyed articles. Pivano et al. [53] and Chowdhury et al. [57], both describe a cloud-based frameworks for digital twins. Chowdhury's work describes a microservice-based architecture in which separate services are deployed and integrated using services like Kubernetes or Docker. In Pivano's work, simulations and data analysis are offloaded to a public cloud server. Doing so grants access to more computational resources and avoids deploying local complex IT infrastructures. Tygesen et al. [50] also notes the increased resources available in the cloud and mentions that high-performance cloud computing is an integrated part of the wave load modeling, a process necessary to maintain an offshore platform's integrity. Zborowski [49] mentions the usage of cloud data lakes where data is verified and fed to physical models and edge-computing-enabled sensors that facilitate data collection, aggregation, and storage, and Brewer et al. [52] believes that the cloud should be used for collaborative data management to reduce costs and increase efficiency while bringing stakeholders into a collaborative digital space for rapid decision making.

8. Conclusion

Before recently, there were few, non-technical definitions of digital twins in the oil and gas (O&G) sector, and there were no universally applicable solutions that could be integrated with those from other suppliers, data sources, and models. Due to the early embrace of digitization, this trend is already further along in other industries. The adoption of digital twins by the O&G industry and the utilization of cloud/edge computing were researched methodically in this article.

For the cloud/edge Digital Twins research in other industries, this adoption is more evolved. Smart Manufacturing were early adopters of digital twins, with the usage of cloud/edge

computing. In healthcare, the focus is on transmission of real-time date for cloud processing, example being the monitoring of patients. In smart vehicles, especially for autonomous vehicles, several researches point to the usage of edge computing, so multiple vehicles can communicate and work together in real-time.

The O&G industries took longer to adopt cloud/edge technologies. Starting with the industry's reluctance to store data outside of their firewall, which could compromise the protection of trade secrets, especially in sensitive areas like well logs, as well as the investment in legacy IT infrastructure that these companies already made in the past, which would require additional costs in migration and raise doubts about whether the investment would be worthwhile. Public cloud computing has shown its dependability over time for various industries, demonstrating that it can be a good alternative in terms of scalability, pricing, and security. From the researched material, the O&G sector has become accustomed to using public clouds, particularly IaaS and PaaS services, to run commercial simulation software on virtualized resources for computationally and storage demanding tasks like seismic processing and reservoir modeling for exploration. For the other scenarios, a transition towards edge computing also become more prevalent, allowing computing to operate in remote regions with unreliable or intermittent network access, without leaving its internal networks.

Divergent views of what constitutes a digital twin for the O&G sector were revealed in the research on digital twins. Some authors present "digital twins" that don't adhere to the entire concept of data collection, processing, and communication of a digital twin, where the real and virtual counterparts converse and develop into a self-contained system. This includes the employing of digital shadows, which take live data from the real system but lack direct communication with it. The majority of the papers chosen in this field concentrate on upstream and drilling. However, the implementations identified place a strong emphasis on fault detection, both for safety and for production, whereas other areas connected to exploration and production do not seem to be as popular in the adoption of digital twins.

Overall, the focus of this paper was to present how cloud and edge computing are key enabling technologies to implement Digital Twins for the Oil and Gas industries, with potential to be used in several different areas. Subjects like the acquisition, storage and sharing of Big Data is an important factor, as collaboration between companies grew together with these tools. Solutions like using the OSDU for sharing production data is a possibility, through cloud services with high availability and with open solutions. Damjanovic-Behrendt and Behrendt [68] presents a microservices-based approach towards the design and implementation of digital twins purely via combining open-source tools, addressing data management, data representation models and services for analytics and optimization. The possibility of running simulations and machine learning on the cloud are possibilities for solving the scenarios expected in the industry, both closed off-the-shelf software from major cloud providers like Google, Amazon and Microsoft and creating customizable alternatives using free open-source software, resolving the issues of privacy and security through the usage of modern, heavily tested solutions while obtaining real-time low latency, via combining it with edge computing and specific solutions like distributed machine learning or federated learning.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported in part by the PETWIN project (financed by FINEP and LIBRA consortium), by the Brazilian National Council for Scientific and Technological Development (CNPq) and in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

[1] D. B. Cameron, A. Waaler, T. M. Komulainen, Oil and Gas digital twins after twenty years. How can they be made sustainable, maintainable and useful?, in: Proceedings of The 59th Conference on Simulation and Modelling (SIMS 59), Linköping University Electronic Press, 2018, pp. 9–16. doi:10.3384/ecp181539.

[2] D. Irving, Causing E&P problems with Digitalisation, in: K. Jeffery (Ed.), Solving E&P problems with digitalisation, Digital Energy Journal, 2018.

[3] P. Parry, Next generation digital organization and capabilities, in: K. Jeffery (Ed.), Solving E&P problems with digitalisation, Digital Energy Journal, 2018.

[4] T. R. Wanasinghe, L. Wroblewski, B. K. Petersen, R. G. Gosine, L. A. James, O. De Silva, G. K. I. Mann, P. J. Warrian, Digital Twin for the Oil and Gas Industry: Overview, Research Trends, Opportunities, and Challenges, IEEE Access 8 (2020) 104175–104197. doi:10.1109/ACCESS.2020.2998723.

[5] F. Tao, H. Zhang, A. Liu, A. Y. C. Nee, Digital twin in industry: State-of-the-art, IEEE Transactions on Industrial Informatics 15 (2019) 2405–2415. doi:10.1109/TII.2018.2873186.

[6] K. M. Alam, A. El Saddik, C2PS: A Digital Twin Architecture Reference Model for the Cloud-Based Cyber-Physical Systems, IEEE Access 5 (2017) 2050–2062. doi:10.1109/ACCESS.2017.2657006.

[7] J. Bönsch, M. Elstermann, A. Kimmig, J. Ovtcharova, A subject-oriented reference model for Digital Twins, Computers & Industrial Engineering 172 (2022) 108556. URL: https://www.sciencedirect.com/science/article/pii/S0360835222005617. doi:https://doi.org/10.1016/j.cie.2022.108556.

[8] R. Balasubramanian, M. Aramudhan, Security Issues: Public vs Private vs Hybrid Cloud Computing, International Journal of Computer Applications 55 (2012) 35–41. doi:10.5120/8818-2808.

[9] R. Narendra, S. Tadapaneni, H. H. Syed, Cloud Computing Security Challenges, International Journal of Innovations in Engineering Research and Technology 7 (2020) 1–6.

[10] A. Avram, FaaS, PaaS, and the Benefits of the Serverless Architecture, 2016. URL: https://www.infoq.com/news/2016/06/faas-serverless-architecture/, acessed: 2022-07-16.

[11] Y. Chen, S. Deng, H. Ma, J. Yin, Deploying Data-intensive Applications with Multiple Services Components on Edge, Mobile Networks and Applications (2020) 426–441. doi:10.1007/s11036-019-01245-3.

[12] W. Kritzinger, M. Karner, G. Traar, J. Henjes, W. Sihn, Digital twin in manufacturing: A categorical literature review and classification, IFAC-PapersOnLine 51 (2018) 1016–1022. doi:10.1016/j.ifacol.2018.08.474, 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018.

[13] F. P. Knebel, An open Digital Twin framework based on microservices in the cloud, Bachelor's thesis, Federal University of Rio Grande do Sul, 2020. doi:10183/219160.

[14] P. D. U. Coronado, R. Lynn, W. Louhichi, M. Parto, E. Wescoat, T. Kurfess, Part data integration in the Shop Floor Digital Twin: Mobile and cloud technologies to enable a manufacturing execution system, Journal of manufacturing systems 48 (2018) 25–33.

[15] L. Hu, N.-T. Nguyen, W. Tao, M. C. Leu, X. F. Liu, M. R. Shahriar, S. N. Al Sunny, Modeling of Cloud-Based Digital Twins for Smart Manufacturing with MT Connect, Procedia Manufacturing 26 (2018) 1193–1203. doi:10.1016/j.promfg.2018.07.155, 46th SME North American Manufacturing Research Conference, NAMRC 46, Texas, USA.

[16] Y. Liu, L. Zhang, Y. Yang, L. Zhou, L. Ren, F. Wang, R. Liu, Z. Pang, M. J. Deen, A Novel Cloud-Based Framework for the Elderly Healthcare Services Using Digital Twin, IEEE Access 7 (2019) 49088–49101. doi:10.1109/ACCESS.2019.2909828.

[17] R. Martinez-Velazquez, R. Gamez, A. El Saddik, Cardio Twin: A Digital Twin of the human heart running on the edge, in: IEEE International Symposium on Medical Measurements and Applications (MeMeA), IEEE, 2019, pp. 1–6. doi:10.1109/MeMeA.2019.8802162.

[18] W. Li, M. Rentemeister, J. Badeda, D. Jöst, D. Schulte, D. U. Sauer, Digital twin for battery systems: Cloud battery management system with online state-of-charge and state-of-health estimation, Journal of Energy Storage 30 (2020) 101557. doi:10.1016/j.est.2020.101557.

[19] W. Xu, J. Cui, L. Li, B. Yao, S. Tian, Z. Zhou, Digital twin-based industrial cloud robotics: Framework, control approach and implementation, Journal of Manufacturing Systems 58 (2021) 196–209. doi:10.1016/j.jmsy.2020.07.013.

[20] P. Bellavista, C. Giannelli, M. Mamei, M. Mendula, M. Picone, Application-driven network-aware digital twin management in industrial edge environments, IEEE Transactions on Industrial Informatics 17 (2021) 7791–7801. doi:10.1109/TII.2021.3067447.

[21] K. Zhang, J. Cao, Y. Zhang, Adaptive Digital Twin and Multiagent Deep Reinforcement Learning for Vehicular Edge Computing and Networks, IEEE Transactions on Industrial Informatics 18 (2022) 1405–1413. doi:10.1109/TII.2021.3088407.

[22] Y. Lu, S. Maharjan, Y. Zhang, Adaptive Edge Association for Wireless Digital Twin Networks in 6G, IEEE Internet of Things Journal 8 (2021) 16219–16230. doi:10.1109/JIOT.2021.3098508.

[23] G. Cathey, J. Benson, M. Gupta, R. Sandhu, Edge Centric Secure Data Sharing with Digital Twins in Smart Ecosystems, in: 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), IEEE Computer Society, 2021, pp. 70–79. doi:10.1109/TPSISA52974.2021.00008.

[24] R. Dong, C. She, W. Hardjawana, Y. Li, B. Vucetic, Deep Learning for Hybrid 5G Services in Mobile Edge Computing Systems: Learn From a Digital Twin, IEEE Transactions on Wireless Communications 18 (2019) 4692–4707. doi:10.1109/TWC.2019.2927312.

[25] Y. Lu, X. Huang, K. Zhang, S. Maharjan, Y. Zhang, Low-Latency Federated Learning and Blockchain for Edge Association in Digital Twin Em-

powered 6G Networks, IEEE Transactions on Industrial Informatics 17 (2020) 5098–5107. doi:10.1109/TII.2020.3017668.

[26] J. Feblowitz, Oil and Gas: Into the Cloud?, Journal of Petroleum Technology 63 (2011) 32–33. doi:10.2118/0511-0032-JPT.

[27] R. Beckwith, Managing Big Data: Cloud Computing and Co-Location Centers, Journal of Petroleum Technology 63 (2011) 42–45. doi:10.2118/1011-0042-JPT.

[28] R. K. Perrons, A. Hems, Cloud computing in the upstream oil & gas industry: A proposed way forward, Energy Policy 56 (2013) 732–737. doi:10.1016/j.enpol.2013.01.016.

[29] F. Saghir, H. Gilabert, M. Boujonnier, Edge Analytics and Future of Upstream Automation, in: SPE Asia Pacific Oil and Gas Conference and Exhibition, OnePetro, 2018. doi:10.2118/192019-MS.

[30] S. O. Settemsdal, B. Bishop, When to Go with Cloud or Edge Computing in Offshore Oil and Gas, in: SPE Offshore Europe Conference and Exhibition, OnePetro, 2019. doi:10.2118/195758-MS.

[31] P. Tedeschi, S. Sciancalepore, Edge and Fog Computing in Critical Infrastructures: Analysis, Security Threats, and Research Challenges, in: 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), IEEE, 2019, pp. 1–10. doi:10.1109/EuroSPW.2019.00007.

[32] M. Lawan, C. Oduoza, K. Buckley, Proposing a conceptual model for cloud computing adoption in upstream oil & gas sector, Procedia Manufacturing 51 (2020) 953–959. doi:10.1016/j.promfg.2020.10.134.

[33] H. Al-Mascati, A. H. Al-Badi, Critical success factors affecting the adoption of cloud computing in oil and gas industry in Oman, in: 2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC), IEEE, 2016, pp. 1–7. doi:10.1109/ICBDSC.2016.7460365.

[34] M. E. Eldred, A. Orangi, A. A. Al-Emadi, A. A. Ahmad, N. Barghouti, Reservoir Simulations in a High Performance Cloud Computing Environment, in: SPE Intelligent Energy Conference & Exhibition, OnePetro, 2014. doi:10.2118/167877-MS.

[35] A. Khodabakhsh, I. Ari, M. Bakir, Cloud-based Fault Detection and Classification for Oil & Gas Industry, in: International Conference on Data Mining, Workshop on Data Mining for Oil and Gas, 2017, p. 14. doi:10.48550/arXiv.1705.04583.

[36] B. Boguslawski, M. Boujonnier, L. Bissuel-Beauvais, F. Saghir, R. D. Sharma, IIoT Edge Analytics: Deploying Machine Learning at the Wellhead to Identify Rod Pump Failure, in: SPE Middle East Artificial Lift Conference and Exhibition, OnePetro, 2018. doi:10.2118/192513-MS.

[37] S. Tanaka, Z. Wang, K. Dehghani, J. He, B. Velusamy, X.-H. Wen, Large Scale Field Development Optimization Using High Performance Parallel Simulation and Cloud Computing Technology, in: SPE Annual Technical Conference and Exhibition, OnePetro, 2018. doi:10.2118/191728-MS.

[38] X. Yang, O. Bello, L. Yang, D. Bale, R. Failla, Intelligent Oilfield - Cloud Based Big Data Service in Upstream Oil and Gas, in: International Petroleum Technology Conference, OnePetro, 2019. doi:10.2523/IPTC-19418-MS.

[39] R. Hussain, M. Amini, A. Kovalenko, Y. Feng, O. Semiari, Federated Edge Computing for Disaster Management in Remote Smart Oil Fields, in: 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), IEEE, 2019, pp. 929–936. doi:10.1109/HPCC/SmartCity/DSS.2019.00134.

[40] T. Wen, E. Dobson, R. Hvaara, Mesh Learning: A Cloud and Edge–based Computing Network Providing Data–Driven Solutions to the Oil and Gas Industry, in: Offshore Technology Conference Asia, OnePetro, 2020. doi:10.4043/30365-MS.

[41] D. Oikonomou, E. Zabihi Naeini, B. Alaei, E. Larsen, Cloud computing in geoscience: Mysteries, miseries, and benefits, Interpretation 9 (2021) SA17–SA22. doi:10.1190/INT-2020-0103.1.

[42] C. P. Gooneratne, A. Magana-Mora, M. Affleck, W. C. Otalvora, G. D. Zhan, T. E. Moellendick, Camera-Based Edge Analytics for Drilling Optimization, in: 2020 IEEE International Conference on Edge Computing (EDGE), IEEE, 2020, pp. 111–115. doi:10.1109/EDGE50951.2020.00024.

[43] A. P. Pink, C. Fehres, J. D. Pearce, S. Costa, M. Noworyta, The new edge. building and deploying a new state of the art, high speed acquisition, automation and analytics platform for drilling, completions, production, and renewable energy applications, in: Abu Dhabi International Petroleum Exhibition & Conference, OnePetro, 2021. doi:10.2118/207291-MS.

[44] H. Huan, L. Liu, B. Huan, X. Chen, J. Zhan, Q. Liu, A theoretical investigation of modelling the temperature measurement in oil pipelines with edge devices, Measurement 168 (2021) 108440. doi:10.1016/j.measurement.2020.108440.

[45] A. Guimaraes, L. Lacalle, C. B. Rodamilans, E. Borin, High-performance IO for seismic processing on the cloud, Concurrency and Computation: Practice and Experience 33 (2021) e6250. doi:10.1002/cpe.6250.

[46] T. Wen, K. Evers, X. Huang, R. Keyes, An Integrated Platform for IIoT in E&P: Closing the Gap between Data Science and Operations, SPE Annual Technical Conference and Exhibition, 2018. doi:10.2118/191489-MS.

[47] G. Schroeder, C. Steinmetz, C. E. Pereira, I. Muller, N. Garcia, D. Espindola, R. Rodrigues, Visualising the digital twin using web services and augmented reality, in: 2016 IEEE 14th International Conference on Industrial Informatics (INDIN), IEEE, 2016, pp. 522–527. doi:10.1109/INDIN.2016.7819217.

[48] G. Saini, P. Ashok, E. van Oort, M. Isbell, Accelerating Well Construction Using a Digital Twin Demonstrated on Unconventional Well Data in North America, in: SPE/AAPG/SEG Unconventional Resources Technology Conference, 2018, pp. 3264–3276. doi:10.15530/urtec-2018-2902186.

[49] M. Zborowski, Finding Meaning, Application for the Much-Discussed "Digital Twin", Journal of Petroleum Technology 70 (2018) 26–32. doi:10.2118/0618-0026-JPT.

[50] U. Tygesen, M. Jepsen, J. Vestermark, N. Dollerup, A. Pedersen, The

True Digital Twin Concept for Fatigue Re-Assessment of Marine Structures, in: International Conference on Offshore Mechanics and Arctic Engineering, volume 51203, American Society of Mechanical Engineers, 2018, p. V001T01A021. doi:10.1115/OMAE2018-77915.

[51] M. Gholami Mayani, R. Rommetveit, S. I. Oedegaard, M. Svendsen, Drilling Automated Realtime Monitoring Using Digital Twin, in: Abu Dhabi International Petroleum Exhibition & Conference, OnePetro, 2018. doi:10.2118/192807-MS.

[52] T. Brewer, D. Knight, G. Noiray, H. Naik, Digital Twin Technology in the Field Reclaims Offshore Resources, in: Offshore Technology Conference, OnePetro, 2019. doi:10.4043/29231-MS.

[53] L. Pivano, D. T. Nguyen, K. Bruun Ludvigsen, Digital Twin for Drilling Operations – Towards Cloud-Based Operational Planning, in: Offshore Technology Conference, OnePetro, 2019. doi:10.4043/29316-MS.

[54] Q. Min, Y. Lu, Z. Liu, C. Su, B. Wang, Machine Learning based Digital Twin Framework for Production Optimization in Petrochemical Industry, International Journal of Information Management 49 (2019) 502–519.

[55] M. G. Mayani, T. Baybolov, R. Rommetveit, S. I. Ødegaard, V. Koryabkin, S. Lakhtionov, Optimizing Drilling Wells and Increasing the Operation Efficiency Using Digital Twin Technology, in: IADC/SPE International Drilling Conference and Exhibition, OnePetro, 2020. doi:10.2118/199566-MS.

[56] M. G. Shirangi, R. Ettehadi, R. Aragall, E. Furlong, R. May, T. Dahl, M. Samnejad, C. Thompson, Digital Twins for Drilling Fluids: Advances and Opportunities, in: IADC/SPE International Drilling Conference and Exhibition, OnePetro, 2020. doi:10.2118/199681-MS.

[57] K. Chowdhury, A. Arif, M. N. Nur, O. Sharif, A Cloud-Based Computational Framework to Perform Oil-Field Development & Operation Using a Single Digital Twin Platform, in: Offshore Technology Conference, OnePetro, 2020. doi:10.4043/30735-MS.

[58] G. M. Rangel, G. A. do Amaral, K. L. de Souza Neves, G. R. Martins, A. G. Neto, L. A. G. B. Júnior, G. R. Franzini, R. Dotta, E. B. Malta, Methodology for Obtaining a Digital Twin for a FPSO Mooring System, in: 26th International Congress of Mechanical Engineering, 2021. doi:10.26678/ABCM.COBEM2021.COB2021-0749.

[59] E. Priyanka, S. Thangavel, X.-Z. Gao, N. Sivakumar, Digital twin for oil pipeline risk estimation using prognostic and machine learning techniques, Journal of Industrial Information Integration 26 (2022) 100272. doi:10.1016/j.jii.2021.100272.

[60] F. Curina, A. Talat Qushchi, A. Aldany, A Case Study for the Development and Use of a Well Control Simulator as a Digital Twin of a Real Scenario, in: SPE Russian Petroleum Technology Conference, OnePetro, 2021. doi:10.2118/206530-MS.

[61] G. B. Avanzini, K. E. Eriksson, Quality Assurance Framework of Digital Twins for the Oil and Gas Industry, in: OMC Med Energy Conference and Exhibition, OnePetro, 2021.

[62] E. LaGrange, Developing a Digital Twin: The Roadmap for Oil and Gas Optimization, in: SPE Offshore Europe Conference and Exhibition, OnePetro, 2019. doi:10.2118/195790-MS.

[63] F. Laborie, O. C. Røed, G. Engdahl, A. Camp, Extracting Value from Data Using an Industrial Data Platform to Provide a Foundational Digital Twin, in: Offshore Technology Conference, OnePetro, 2019. doi:10.4043/29576-MS.

[64] F. Akbari, Intelligent Digital Twins and Augmented Reality in Inspection and Maintenance, in: Abu Dhabi International Petroleum Exhibition & Conference, OnePetro, 2021. doi:10.2118/207659-MS.

[65] M. M. Said, R. Pilgrim, G. Rideout, S. Butt, Theoretical Development of a Digital-Twin Based Automation System for Oil Well Drilling Rigs, in: SPE Canadian Energy Technology Conference, OnePetro, 2022. doi:10.2118/208902-MS.

[66] K. Abdrakhmanova, A. Fedosov, K. Idrisova, N. K. Abdrakhmanov, R. Valeeva, Review of modern software complexes and digital twin concept for forecasting emergency situations in oil and gas industry, in: IOP Conference Series: Materials Science and Engineering, volume 862, IOP Publishing, 2020, p. 032078. doi:10.1088/1757-899x/862/3/032078.

[67] M. G. Mayani, M. Svendsen, S. I. Oedegaard, Drilling Digital Twin Success Stories the Last 10 Years, in: SPE Norway One Day Seminar, 2018. doi:10.2118/191336-MS.

[68] V. Damjanovic-Behrendt, W. Behrendt, An open source approach to the design and implementation of Digital Twins for Smart Manufacturing, International Journal of Computer Integrated Manufacturing 32 (2019) 366–384. doi:10.1080/0951192X.2019.1599436.

APPENDIX C — REDES DE COMPUTADORES APRIMORADAS POR SDN E NFV COM O USO DE GÊMEOS DIGITAIS (PORTUGUESE)

The following appended document is the final works for the CMP610 course, Software-based and Programmable Networks, written in Portuguese. The paper features a systematic research combining computer networks, SDN (Software Defined Networking), NFV (Network Function Virtualization) and Digital Twins. Its main goal is defining the state of the art of the digital twin research in the context of programmable networks. From the 200 most relevant results from two queries made in Google Scholar, a total of 67 papers were analyzed and included in the final results. The article list generated 9 different research topics, serving as a brief summary of the contributions in the three key topics of the research, and how they interact with each other for computer networks.

Redes de Computadores aprimoradas por SDN e NFV com o uso de Gêmeos Digitais

Francisco Paiva Knebel
Instituto de Informática
Universidade Federal do Rio Grande do Sul
Email: francisco.knebel@inf.ufrgs.br

I. Introdução

Um gêmeo digital é uma representação virtual e digital de um sistema real físico, podendo ser utilizado para prever estados futuros da entidade física. O gêmeo digital é integrado com outras tecnologias, como inteligência artificial, mineração de dados, computação em nuvem e a Internet das Coisas (*Internet of Things*, IoT). Através de protocolos de comunicação, o gêmeo digital e seu equivalente físico estão intrinsecamente conectados, gerando um fluxo de informações em tempo real dos sensores físicos ao gêmeo, e dele para atuadores do sistema físico. Isso permite a coleta de informações do funcionamento do sistema e a sugestão por parte do gêmeo de ações sobre o sistema real, permitindo um produto mais eficiente e criando inteligência pela análise dos processos efetuados.

A arquitetura de um gêmeo digital implica na criação de uma réplica digital de algo, sendo ela uma pessoa ou objeto quaisquer, em um ambiente virtual e então conectando ambos através de uma conexão de rede, permitindo a troca de informação em tempo real entre os gêmeos. Implementando-se em um sistema industrial, uma máquina conectada poderia ser modificada por humanos interagindo com seu gêmeo digital, com operações sendo despachadas do ambiente virtual para o gêmeo real [1]. Um gêmeo digital de um componente é uma entidade de software que espelha outro componente, podendo ele ser um sistema ciberfísico (CPS, *Cyber Physical System*), como um sensor ou uma linha de produção, até um processo de produção ou uma fábrica inteira. Esses componentes digitais podem ser utilizados para simular e testar a operação de um produto antes de comprometer o sistema real a funcionar da mesma forma [2].

Com este trabalho, é pretendido obter o estado da arte de Gêmeos Digitais para Redes de Computadores, com ênfase em trabalhos quem implementem ou discutam possíveis soluções no contexto de redes programáveis baseadas em software, utilizando as tecnologias de SDN (*Software Defined Networking*) e NFV (*Network Functions Virtualization*).

O trabalho está separado da seguinte forma: na seção II está descrito a metodologia de pesquisa, como foram obtidos os trabalhos e os critérios de inclusão e exclusão utilizados; na seção III, o conteúdo da pesquisa é apresentado, separado pela classificação dos temas dos artigos; e na seção IV é efetuada a conclusão sobre os resultados obtidos.

II. Metodologia

Por se tratar de uma pesquisa sistemática, foram seguidas regras estritas para a seleção de trabalhos. Como ferramenta de busca, o levantamento de artigos foi feito com o auxílio do Google Scholar, efetuando a busca manualmente pelos principais artigos retornados para cada *query* de pesquisa. Uma série de pesquisas foram efetuadas separadamente, de forma a incluir trabalhos que cobrem suficientemente o assunto da pesquisa. Em seguida, após a obtenção dos trabalhos, foi feito a classificação dos artigos para incluir nesta pesquisa.

A. Pesquisa

O objetivo inicial da pesquisa é definir o estado da arte da pesquisa de gêmeos digitais nos contextos de redes programáveis, em específico sobre as contribuições que utilizam os conceitos de SDN e NFV. Efetuando um corte dos trabalhos que relacionam essas três palavras-chave, conforme a Fig. 1, podemos obter de qual forma o uso de SDN e de NFV estão sendo usados no contexto de gêmeos digitais para a criação e manutenção de redes de computadores.

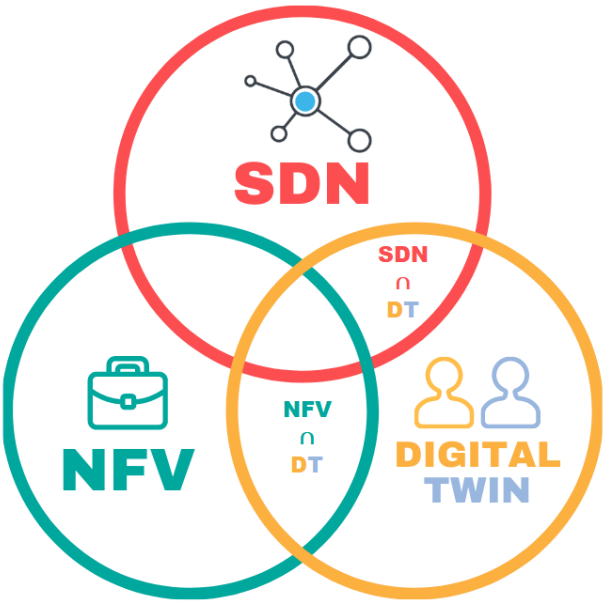


Fig. 1. Assuntos-chave da pesquisa.

Para obter cada um dos subgrupos de artigos utilizados como referência para este estudo, foi efetuado uma série de pesquisas via Google Scholar, utilizando o seu algoritmo próprio para classificar a relação da pesquisa com a qualidade do artigo. Todas as pesquisas foram efetuadas em Abril de 2022 e não efetuaram corte de artigos pelo período de publicação. Os seguintes termos foram utilizados:

1) **"Digital Twin" SDN**: inclusão de trabalhos que ligam diretamente gêmeos digitais com SDN. Obtido 980 resultados.
2) **"Digital Twin" NFV**: inclusão de trabalhos que ligam diretamente gêmeos digitais com NFV. Obtido 425 resultados.

Cada uma das pesquisas efetuadas retornou uma quantidade considerável de artigos para seleção, o que gera uma preocupação com a obtenção de artigos suficientes. Por não se tratar de uma pesquisa sistemática exaustiva, foram obtidos os primeiros 100 artigos para cada pesquisa, utilizando a ordenação por relevância do próprio Google Scholar como critério de exclusão. Dessa forma, por se tratarem de duas pesquisas, foram obtidos uma lista de 200 artigos para serem classificados.

B. Classificação

Com a lista de artigos obtida, várias etapas de classificação foram efetuadas para reduzir a lista de artigos que seriam incluídos na leitura manual. Os seguintes critérios de exclusão foram utilizados, efetuando análise manual sobre o conjunto de artigos:

1) **Exclusão por duplicata**: Artigos que apareceram em mais de uma das pesquisas, além de artigos que aparecem repetidos dentro de uma pesquisa. Exclusão de 43 artigos.
2) **Exclusão por linguagem**: Artigos publicados em outra linguagem além da língua inglesa. Exclusão de 7 artigos.
3) **Exclusão por acesso ao artigo**: Artigos que não puderam ser livremente obtidos ou através de distribuidoras com fácil acesso foram excluídos. Na maioria dos casos, os resultados são livros ou capítulos de livros. Exclusão de 13 artigos.
4) **Exclusão por falso-positivo**: Filtro manual, efetuado durante leitura final da lista, por entradas que apenas mencionam as palavras-chaves, mas que não apresentam contribuições, além de casos óbvios de inclusão incorreta feita pelo Google Scholar, como trabalhos de outro contexto, não de redes de computadores, fora do contexto da pesquisa. Exclusão de 70 artigos.

Dessa forma, da lista inicial de 200 artigos incluídos pela pesquisa, 133 foram removidos devido aos critérios de exclusão. Deste montante, restaram após a filtragem por todos critérios um total de 67 trabalhos incluídos na pesquisa final[1].

[1]Dados da pesquisa, incluindo a lista completa e cada etapa do processo de exclusão podem ser vistos em uma planilha pública, disponível em: https://github.com/Open-Digital-Twin/article-sdn-nfv-digital-twins/raw/main/TF-Research.xlsx

III. Pesquisa

Com a leitura dos artigos, foram definidas múltiplas categorias de trabalhos, sendo o seu conteúdo resumido nas subseções seguintes e seus artigos classificados.

A. Gêmeos Digitais para Redes

O IETF (Internet Engineering Task Force) possui um grupo atualmente trabalhando na padronização do conceito de uma plataforma de gêmeos digitais para redes [3], emulando os elementos da rede digitalmente. A principal diferença entre sistemas tradicionais de gerenciamento é a interatividade virtual e real, digital e físico, construindo um sistema fechado e automatizado, que se comunica para aprender. Através da integração de dados em tempo real da rede física com sua equivalente digital, a plataforma pode ser simplificada, se tornar mais resiliente e efetuar manutenção sobre todo o ciclo de operação. O gêmeo digital construído possibilita a rápida detecção e solução de problemas na rede, predição de status futuro da rede e melhora na sua confiabilidade por eliminar riscos conhecidos antes de acontecerem.

Digitalização de um sistema físico para o gêmeo digital necessita de desacoplamento de informação e de funções. Desacoplamento de informação (ou desacoplamento de dados) permite a representação do estado geral do sistema, efetuando o design do sistema de forma independente dos dispositivos de rede [4]. Desacoplamento de função se refere a alocação de recursos, de administração de mobilidade, desacoplando as funções de gerenciamento da camada física para a camada virtual. Pode ser implementado pelo fatiamento de rede baseado em NFV e SDN, separando o plano de controle da interação física para facilitar a administração das funções de rede. O uso de SDN simplifica e torna mais eficiente a elevação de processos mais complexos, que necessitam de supervisionamento e administração dinâmica de recursos, requerendo reconfiguração dos elementos de rede [1].

Um gêmeo digital separa as funções de controle, implantadas como um sistema logicamente centralizado, dos dispositivos físicos sob controle, de forma similar ao SDN. Em uma arquitetura de rede baseada em SDN, o plano de controle é separado do plano de dados, permitindo ao SDN centralizar o processo de controle e permitir a implementação de uma rede programável, dinamicamente alterando-se com as demandas de tráfego exigidas [5], além de expôr visibilidade total das configurações da rede e o seu estado, além de efetuar o controle de fluxos [6]. O controlador SDN coleta informações sobre todo o sistema interconectado, podendo ser o local de formulação de estratégias para garantir o seu funcionamento [7], além de fornecer informações e a conexão necessária para aprendizado sobre o estado global da rede [8]. Da mesma forma, gêmeos digitais permitem a realização de tomada de decisões e controle de processos em um ambiente centralizado, separando o processo de controle (no ambiente virtual) do plano de dados (ambiente real), definindo os fluxos e interações entre os dispositivos conectados e sua interação com o ambiente.

Outro aspecto de gêmeos digitais é a virtualização de objetos e processos físicos, uma visão paralela à NFVs, que permitem a implementação de funções de redes em forma de software, que pode ser executado em máquinas virtuais genéricas, ao invés de depender de equipamentos especializados, o que flexibiliza o desenvolvimento de novas funcionalidades para a rede. Gêmeos digitais estendem o conceito para a virtualização de qualquer função lógica vinda de objetos ou processos.

Uma rede implementada com gêmeos digitais permite o desenvolvimento de soluções com otimizações ótimas, solucionando problemas, efetuando análise de cenários e planejando melhoras na rede levando em consideração simulações do crescimento esperado. Além disso, todos esses processos podem ser efetuados sem prejudicar a rede física, ocorrendo puramente no ambiente virtualizado. Operadores de rede podem reproduzir falhas anteriores, para descobrir a origem da disrupção de serviço, além de auxiliar em soluções para prevenir disrupções futuras. Estudos sobre gargalos, más configurações de rede, observação de performance em caso de perda de *links* e detecção automática de anomalias são alguns benefícios fornecidos por uma implementação baseada em gêmeos digitais [9].

Em um ambiente NFV, detecção, diagnóstico e resolução de componentes falhos na rede dependem de intervenção humana, mas com o crescimento e diversificação das redes, como pela customização de funções virtualizadas de rede (VNF, *Virtualized Network Function*) baseadas em requisições dos usuários para gerarem funções personalizadas, soluções para resolução de problemas operacionais de forma autônoma são necessárias. Para resolver esse problema, algumas soluções são sugeridas: detecção de anomalias para encontrar padrões em dados que não são esperados; análise de causa raiz (Root Cause Analysis, RCA) serve para encontrar componentes falhos causando anomalias e encontrar as causas para degradação de serviço, como latência, que podem ser causadas por diferentes razões; e compensação, para executar ações e recuperar a rede para seu estado anterior, antes da falha, após corrigido. Para obter a dependência de uma relação com falhas anômalas e implementar essas soluções, um gêmeo digital pode ser introduzido, continuamente se atualizando com o ciclo de vida da rede. Através do monitoramento da rede física, o gêmeo digital é capaz de capturar a dependência anômala, além de poder facilmente criar e transferir o conhecimento para uma nova instância dessa dependência [10]–[12].

B. 5G, 6G e Sistemas Wireless

SDN oferece a separação do plano de controle do plano de dados. Em um plano de controle típico de SDN, um único controlador centralizado é usado para controlar os múltiplos *switches*, que executam diferentes funções [13], [14]. Controladores formam o plano de controle, tomando controle da rede e comandando os dispositivos do plano de dados. Eles também coletam informações da rede física, e um gêmeo digital acoplado pode tomar vantagem dessa conexão para efetuar monitoramento da rede, efetuar manutenção preditiva e diagnosticar a rede. O funcionamento dependente de um único controlador centralizado pode sofrer com problemas de escalabilidade e confiabilidade, além de que o aumento de dispositivos irá causar um aumento de latência, por isso é sugerido uma implementação híbrida, contendo controladores centralizados e distribuídos, onde o controlador centralizado é responsável pelos controladores locais [4]. NFV oferece implementação eficiente e de custo-beneficio de funções de rede usando máquinas virtuais operando em hardware genérico [15].

O fatiamento de rede é uma estratégia proeminente para manter a mesma infraestrutura disponível para múltiplos operadores. Ele funciona através da virtualização dos recursos de rede e alocação de partes dele (logo o termo fatia) para cada parte interessada. SDN permite a administração dos recursos através de suas políticas implementadas, ao mesmo tempo que NFV desacopla as funções de rede dos dispositivos físicos que executam tais funções para implantação em máquinas virtuais [16]. As fatias de rede são isoladas logicamente e redes virtualizadas compartilhando uma infraestrutura física comum podem ser administrados e controlados para suportas um provisionamento de serviços flexível, afim de satisfazer as necessidades dos usuários [17]. Fatiamento de rede pode ser utilizado para prover recursos sob medida para a indústria 4.0, mas sua implementação, devido ao aumento de complexidade da rede é um desafio. A rede é dividida em fatias de rede baseadas nos serviços que ela suporta, mantendo a mesma infraestrutura física, que operam de forma independente e podem ser otimizados para a utilização para os requisitos de um usuário específico [18]. SDN e NFV decompõem o formato monolítico e proprietário de redes tradicionais em módulos menores chamados VNFs (Virtual Network Functions), sendo executadas em hardware genérico ao invés de *switches* dedicados. Um desafio de computação distribuída é criada pelo fatiamento: como acomodar múltiplas fatias de rede utilizando os mesmos recursos de rede. Exemplos são o isolamento de conectividade da fatia, ou seja, usuários de uma fatia não podem se comunicar com serviços de outras fatias, e de isolamento de performance, onde uma fatia não deve afetar o funcionamento de outra [19]. O escalonamento automático de VNFs, para lidar com a demanda, introduz vantagens em menor custo de implantação e taxa de insatisfação das requisições pelos serviços, além de maior capacidade de resiliência para falhas de hardware. Entretanto, sistemas distribuídos introduzem complexidade para a computação, podendo gerar inconsistência de dados e gerando problemas de sincronização [20].

Larsson [21] implementa virtualização da rede utilizando P4, avaliando em termos de escalabilidade de redes distribuídas, considerando cada rede como um gêmeo, que pode estar distribuído através de diferentes locais, conectados num *backbone* comum. Os resultados da implementação em P4 são comparados com implementações não-baseadas em P4, de trabalhos relacionados. Performance das diferentes tecnologias são avaliadas, e o autor considera duas soluções para virtualização da rede: uma com aprendizado baseado no

plano de dados e outra com aprendizado baseado no plano de controle. O nível de programabilidade fornecido por iniciativas como o P4 facilita a implementação de novos protocolos rapidamente, mas que não é possível concluir que o uso de uma tecnologia ou outra para virtualização de redes distribuídas é a melhor, pois dependo do contexto da aplicação, mas que a implementação em P4 permitiu grande escalabilidade sem *switches* físicos.

O uso de gêmeos digitais pode ser efetuado para criar uma representação virtual de redes fatiadas, afim de simular seus comportamentos e prever sua performance. A implementação é feita através de nodos e link, que podem ser divididos e isolados em contêineres e links virtuais, contendo VNFs. Gêmeos digitais podem beneficiar o gerenciamento das fatias de rede pela criação das representações virtuais dos pedaços físicos da rede, podendo ser usado para simular cenários sem afetar a rede real. [22]. Para tirar vantagem do fatiamento de rede, é crítico o monitoramento eficiente da rede, com geração de métricas suficientes para efetuar o gerenciamento autônomo e a orquestração dinâmica da rede para garantir os requisitos de QoS (Quality of Service) das aplicações. Entretanto, a automação dessa garantia representa desafios adicionais na orquestração de serviços, com a alocação dinâmica de recursos e escalonamento automático. Para garantir que as fatias suportem os requisitos dos serviços, os provisores terão que implementar inteligência nos serviços e na rede, além de conhecimento do contexto por parte da fatia [23].

A implementação do 5G ainda é muito recente e introduz muitos desafios. Com o suporte de inteligência artificial, o uso de gêmeos digitais para redes 5G tem potencial de facilitar e completar a sua implementação. NFV e SDN permitem a flexibilidade de posicionamento das funções de rede, na borda ou na nuvem, implantando apenas as funções necessárias [2], [24]. Requisitos de indústrias verticais do 5G[2] muitas vezes são contraditórios, impondo latências extremamente baixas ao mesmo tempo que outros demandam taxas de banda extremamente altas, tornando difícil a definição de redes de propósito genérico [25], [26]. Podemos extrapolar o conceito de gêmeos digitais em redes até o futuro 6G, onde taxas de dados extremamente altos com baixíssima latência serão a norma. O conceito de desacoplamento será essencial para a implementação do 6G. A combinação de SDN e NFV são fundamentais para o fatiamento da rede, mas o fatiamento necessário para o 6G será diferente: gêmeos digitais enriquecidos pelo 6G usarão uma representação digital do sistema físico, utilizando aprendizado de máquina para proativamente analisar e modelar as funções de rede. Junto de NFV, o fatiamento da rede será mais granular, otimizando o acesso a recursos [27]. Um gêmeo digital do 6G irá depender diretamente do fatiamento de rede, desacoplamento de dados, análise proativa e de otimização dos recursos fornecidos pela introdução de inteligência dentro da rede, que só poderá ocorrer com a introdução de programabilidade, de forma a obter uma orquestração de rede completamente autônoma [2], [13], [26], [28], [29]. Ao mesmo tempo que o 6G pode facilitar a realização e adoção de gêmeos digitais em múltiplas industrias, provendo os níveis necessários de confiabilidade e velocidade, gêmeos digitais com seu poder de inteligência artificial podem facilitar o design, implantação e operação do 6G. Uma implementação que leva em conta esses critérios pode ter alto impacto para atingir uma rede de alta confiabilidade [30]. Gêmeos digitais também podem ser utilizados para melhor o fatiamento da rede, provendo dados organizados e customizados para as fatias, refinando a abstração das fatias para um nível maior de personalização [31].

Redes programáveis, baseadas em software através de SDN e NFV, é uma das principais tecnologias facilitadoras para o 6G. Com a virtualização, o desacoplamento de serviços e permitindo maximizar o uso da rede entre os diferentes serviços usando a mesma infraestrutura possibilita que os provedores de serviço compartilhem dinamicamente da mesma rede de física que os operadores de rede móvel. O uso de SDN permitirá a reconfiguração dinâmica da topologia de rede de acordo com a demanda e adicionar mais recursos de rede para manter a qualidade de serviço para os usuários. Para os operadores de rede, isso significa uma rede mais fácil de monitorar e manter, com redução de CapEx (*Capital Expenses*, custo capital) e OpEx (*Operational Expenses*, custo operacional), além de inovação mais rápida, pela facilidade de implantação em software ao invés de hardware [32]–[34]. Para a implantação das redes 6G, é esperado que arquiteturas de rede baseadas em software utilizando infraestrutura definida em software seja amplamente utilizada [35].

Para a implementação de gêmeos digitais, o uso de tecnologias como computação de borda e NFV praticam um papel importante para o oferecimento de baixa latência, resiliência, alta banda e escalabilidade [36], [37]. *Multi-Access Edge Computing* (MEC) é um paradigma que combina esses elementos para que operadores abram acesso à rede para serviços tirarem vantagem da grande proximidade com o usuário, facilitado por NFV pela redução de custos de hardware através da virtualização, o provisionamento flexível (escalar para cima/baixo de acordo com a demanda) de recursos e a rápida instanciação de novos serviços. O *offloading* de funções para a computação na borda introduz reduções no desperdício de computação e uso de memória [38], [39].

Introduzindo serviços de rede altamente flexíveis automatizam muito do processo manual de configuração, mas a introdução de NFV em sistemas industriais ainda apresenta alguns desafios, como a integração de protocolos legados ainda usados em comunicação de máquinas existentes, que são consideravelmente diferentes de protocolos mais recentes, como REST, além da dificuldade de lidar com a complexidade de diferentes serviços virtualizados [40]. Além disso, implementar um gêmeo digital pode não ser a solução mais eficiente para muitos provedores, pois gera computação extra e um ponto adicional de ameaça para a segurança do sistema. Um cenário de aprendizado distribuído, como é o caso, causa custos de

[2]Um vertical da indústria é um termo utilizado para definir grupos de empresas com foco em um nicho específico ou com mercado especializado que abrange várias indústrias. Mais informações em "https://web.archive.org/web/20220311194057/https://pitchbook.com/what-are-industry-verticals".

armazenamento adicionais para a etapa de treinamento [41].

Uma proposta híbrida de SDN, adicionando um gêmeo digital, é proposta por Taylor [42] para adicionar mais funcionalidades para suportar grandes quantidades de nodos. O controlador centraliza a configuração de dispositivos de rede e monitoramento, e o gêmeo digital virtualiza os elementos na borda, recomendando configurações para o plano de controle. Para redes *wireless*, o plano de controle é um híbrido entre os planos de controle e dados, pois os dados de controle do *gateway* e do ponto de acesso podem ser transmitidos sem-fio junto com os pacotes de dados. Um gerenciador de topologia é implementado dentro do sistema, coletando dados de geolocalização dos nodos da rede, afim de se autoconfigurar, pela modificação dos nodos.

C. Internet das Coisas

Virtualização é um conceito diretamente ligado com a implementação de gêmeos digitais. Em redes, virtualização é baseado em melhor explorar o hardware disponível, executando diferentes aplicações utilizando o mesmo hardware genérico. Esse processo está normalmente acoplado com SDNs, efetuando o desacoplamento do hardware da rede do seu software controlador. Orquestração é fundamental para coordenar a alocação dos recursos, ou seja, computação, armazenamento e recursos de rede dentro da infraestrutura virtualizada. Virtualização da rede oferece a capacidade de lidar com a virtualização de funções, sua orquestração e o encadeamento entre diferentes serviços à infraestrutura do gêmeo digital [43].

A combinação de IoT com o ecossistema industrial é uma possibilidade de implementação de gêmeos digitais, permitindo redução de custos em equipamentos e manutenção, monitoramento de recursos, otimização de manutenção, economia de energia, além de possibilitar a conexão entre dispositivos inteligentes. SDN podem dinamicamente refletir os requisitos de comunicação necessários, levando em consideração as condições de rede, sendo um mecanismo de controle da importância das mensagens e do estado geral da rede, de forma a proporcionar a melhor qualidade de serviço possível definindo as regras de encaminhamento de cada elemento da rede. O controle efetuado define o caminho para as mensagens do gêmeo digital para garantir as condições de tempo-real necessárias [44]–[46]. A integração de tecnologia industrial, gêmeos digitais, SDN e NFV são habilitadoras para mover o processamento para a computação em borda. Fusão, aquisição e mineração de dados serão essenciais para essa implementação nos espaços industriais, mas apresentam desafios pela escala massiva de sistemas e tipos de dados a serem modelados até a integração com sistemas de aprendizado [47].

O uso de SDN para dinamicamente modificar como os nodos na borda administram os fluxos de tráfego, lidar com configurações e requisitos dinâmicos de QoS, como tolerância de latência, com capacidade de adaptação da rede. Metadados podem ser analisados pelo controlador para priorização de mensagens. SDN emergiu primariamente para administrar *switches* em *datacenters* fechados e centralizados, mas sua adoção provou seus benefícios em cenários com poder computacional e de capacidade de rede mais limitados. SDNs podem ser usados para manter os requisitos de latência e garantir QoS na troca de dados em tempo real, atingindo melhor sincronização e confiabilidade entre sistemas [48].

Focando em ambientes industriais, SDN em IIoT é primariamente usado em ambientes fechados, utilizando OpenFlow. SDN pode ser adotado para dinamicamente explorar os mecanismos de comunicação mais adequados para os requisitos da aplicação, como utilizando dados sobre o contexto do pacote para mais eficientemente definir regras de fluxo de tráfego [49]. Um gêmeo digital interligado com a rede pode efetuar o gerenciamento da rede internamente ao sistema, com o controlador da rede softwarizada conectando o gêmeo digital de cada parte do sistema industrial com o seu parceiro físico [14]. A precisão e tempo de reação de gêmeos digitais permitem o controle remoto por operadores humanos de robôs industriais a longas distâncias [50].

Para lidar com dispositivos em movimento, Santa et al [51] propõem que para a continuidade de processamento seja o objetivo, a transferência dos recursos de processamento e o estado da tarefa sendo processada deve ser efetuada para manter o serviço. A sua solução utiliza o conceito de *virtual mobile devices* (vMDs) como funções virtualizadas representando dispositivos físicos, instanciados na borda da rede, utilizando SDN para migrar os dispositivos ao longo da rede, sem perda de computação, através da transferência de estados de gêmeos digitais com capacidade de processar dados para dispositivos em movimento de forma transparente.

Redes Sensíveis ao Tempo (TSN, *Time Sensitive Networking*) é um tecnologia criada para a implementação de redes determinísticas com requisitos de tempo real, o que inclui aumento de confiabilidade, controle de latência, sincronização de relógio e gerenciamento de recurso, necessários para manufatura avançada assistida por robótica. Ter esses bens industriais virtualizados dependem do fatiamento de rede e assistência fornecidas por SDN e um controlador centralizado para configuração da rede. Gêmeos digitais podem ser usados nesse contexto para otimizar a imprecisão da rede, com seus requisitos dinâmicos, calculando as configurações de roteamento e agendamento, além de manter um modelo atualizado da rede, com constante atualização vinda pelos dados de telemetria [52], [53].

A aplicação de SDN para redes de sensores IoT tem o desafio de lidar com a baixa capacidade dos dispositivos conectados, tanto computacional quanto de comunicação. O uso de virtualização das funções de rede e para criar a representação virtual dos sensores dependem de NFV, possibilitando o enriquecimento dos dispositivos com mais recursos, como capacidade adicional de computação através do pré-processamento de dados, de comunicação através de maior variedade de protocolos de transmissão, diferentes dependendo de cada aplicação ou de armazenamento com dados coletados pelos sensores sendo pré-processamento localmente antes de serem transportados para um coletor de dados, pendente a disponibilidade de conectividade [2].

D. Redes Veiculares

Essa seção se trata da combinação de SDNs e gêmeos digitais no contexto de redes veiculares, que oferecem conectividade entre veículos e infraestrutura em rede, oferecendo aplicações e serviços. VANETs (*Vehicular Ad-hoc NETworks*) tradicionais possuem o desafio de como possibilitar uma rede inteligente com seu formato descentralizado, onde um veículo não possui capacidade de coletar e computar grandes quantidades de dados. A implementação de SDN para redes veiculares introduz com o controlador o poder computacional não presente em veículos individuais, tendo a visão global da rede e podendo adaptar de forma inteligente o roteamento para a demanda dinâmica desse ambiente [54].

Gêmeos digitais também são essenciais para a implementação de veículos autônomos. A transformação para veículos com operação baseada em dados torna necessária a adoção de novas regras para aumentar a resiliência e segurança dos veículos, essenciais para a redução de acidentes e para manter o cuidado num ambiente volátil, que envolve motoristas de outros veículos e pedestres. Veículos autônomos estão sujeitos a múltiplos tipos de falhas: por falha do sistema, sendo de origens mecânicas, eletroeletrônicas, ou falhas por ataques, tanto física quanto por ciberataques [55].

Redes VEC (Vehicular Edge Computing), criadas para melhor prover aplicações e serviços em proximidade de veículos, reduzem a latência de transmissão e o congestionamento da rede. Entretanto, desafios para implementação de redes como essa são a alta mobilidade de veículos, num ambiente dinâmico, necessitando administração de alta complexidade. Auxiliadas por gêmeos digitais, redes VEC podem adaptativamente administrar os recursos da rede e o cronograma de políticas de encaminhamento. Em tempo real, o gêmeo digital pode monitor os estados dos veículos e os recursos de rede, obtendo uma análise precisa da rede. Além disso, ele criará a camada virtual entre as entidades físicas da VEC e as aplicações veiculares, criando a ponte entre aplicações e usuários. Por ser responsável por essa comunicação, todo o fluxo de dados passa pelo gêmeo, sendo que dessa forma a rede VEC pode se adaptar para mudanças dinâmicas na topologia da rede, se adaptar a mudança de localização dos veículos e se adaptar para situações de emergência, alterando o cronograma de políticas e efetuando *offloading* de tarefas [56].

Zaid [57] argumenta que SDN, NFV e gêmeos são tecnologias habilitadoras para a implementação de eVTOLS (de *electric Vertical Take-off and Landing*, veículos urbanos para transporte aéreo), afim de ser uma alternativa ao trânsito terrestre tanto para o transporte de bens e produtos quanto para locomoção de passageiros. Ainda se tratando de uma tecnologia futuro, é possível especular alguns aspectos básicos necessários para o seu funcionamento e sua implementação, como comunicação confiável de alta velocidade e de baixa latência, com coordenação colaborativa entre os veículos, pela importância da carga carregada e para dirigir de forma autônoma. Para redes alcançarem esses objetivos, é essencial a introdução de reprogramabilidade da rede, dependência de um controlador centralizado para coordenação dos veículos e virtualização de componentes para serem escalados dentro da rede. Gêmeos digitais de eVTOLS serão utilizados nesse contexto, obtendo uma visão geral do sistema, afim de garantir a segurança e o planejamento de rotas [57].

E. Redes Ópticas

Uma rede óptica é uma tecnologia de rede altamente adotada por prover alta taxa de dados com baixo custo de operação comparada com outras tecnologias, mas possui a desvantagem de alto consumo de energia. O design de uma rede eficiente e inteligente pode levar em conta o comportamento de uso da rede pelos consumidores, utilizando aprendizado de máquina para ser mais eficiente no consumo de energia. Uma rede baseada em SDN tem o benefício do monitoramento de dados fornecido pelo controlador, regularmente coletando dados de toda infraestrutura, como estatísticas do tráfego e *logs* de eventos. Como conservação de energia e manter o maior QoS possível são dois objetivos contraditórios, o poder de decisão e o monitoramento próximo das operações devem ser executados de forma reativa e proativa em ordem para minimizar o consumo sem deteriorar o QoS [58].

Alabarce et al [59] discute o uso de gêmeos digitais como habilitador de redes sem toque (*Zero Touch Networking*), redes com a capacidade de atualização, provisionamento e *upgrade* das capacidades da rede por meio de automação com o mínimo de intervenção humana, reduzindo custos de manutenção. Isso é possível através de diagnósticos automáticos e validação de corretude das configurações de rede antes de sua implementação em campo, ou seja, no contexto de um gêmeo digital de redes, simular o funcionamento da rede modificada no ambiente digital antes de sua implantação no ambiente físico [59]. Gêmeos digitais podem ser utilizados para prover flexibilidade experimental, antes da implementação da rede [60], e seu uso de aprendizado de máquina e mecanismos de inteligência artificial é essencial para a existência de redes sem toque.

F. Segurança

A introdução de sistemas como gêmeos digitais introduz o potencial de ataques por meios digitais. Gêmeos digitais serão utilizados na crescente digitalização de indústrias de manufatura, construção, cidades, saúde, logística e energia. Apesar de todos os benefícios gerados para esses sistemas, altamente ligados à tomada de decisões de sistemas inteligentes. Devido à escala de comunicação e a interação com os seus parceiros reais, manter a segurança desse tipo de tecnologia é ainda mais essencial, pois pode causar efeitos diretos no mundo físico, vulnerável ao uso malicioso. A implementação segura de gêmeos digitais para infraestruturas heterogêneas é essencial. Infraestruturas inteligentes baseadas em SDN podem ser utilizadas para que redes de comunicação se tornem mais resilientes a ataques. Caminhos redundantes permitem o aumento da resiliência da rede e estratégias para mitigar ataques a comunicação de dados podem ser explorados, como

técnicas de criptografia, que torna a comunicação mais segura contra muitos tipos de ataques [61].

Ameaças a segurança de gêmeos digitais são variadas, podendo ser ataques a comunicação do sistema e à forma de armazenamento de dados. Desafios de segurança, como modificação não autorizada de informação do gêmeo, interfaces de comunicação mal configuradas, autenticação vulnerável são algumas possibilidades [4]. Outro tópico de interesse é confidencialidade, pois vazamentos de dados podem ser críticos ao funcionamento de sistemas, contendo problemas como propriedade intelectual e segredos comerciais. Karaarslan [61] classifica para gêmeos digitais sete grupos de ameaças à segurança:

- Ameaças físicas: segurança dos dispositivos físicos, que podem ser danificados, destruídos ou modificados.
- Ameaças de modificação de dados: modificação de dados irá passar informação incorreta para o gêmeo, gerando problemas para as previsões do sistema. Devido à comunicação direta com esses dispositivos, é necessário cuidado extra com o ambiente físico, pois um dispositivo infectado poderá afetar o comportamento de todo o sistema.
- Ameaças sistêmicas: ataques ao sistema operacional do gêmeo, que podem tomar controle sobre a sua operação, gerando problemas como negação de serviço, além de ataques maliciosos direcionados a atacar sistemas industriais.
- Ameaças de software: permitir acesso não autorizado ao gêmeo irá informar um atacante sobre o estado geral do sistema, que poderá analisar e buscar por vulnerabilidades. Um acesso dessa forma poderia ser obtido por violações de segurança por terceiros, inserindo código malicioso dentro do sistema.
- Ameaças de comunicação de dados: gêmeos digitais dependem de uma grande quantia de componentes heterogêneos serem integrados. Ameaças sobre comunicação, como *Denial of Service*, *Eavesdropping*, *Spoofing* e *Man-in-the-middle* são alguns cenários possíveis.
- Ameaças de armazenamento de dados: dados de operações para gêmeos digitais normalmente são armazenados em um cenário de nuvem. A centralização de armazenamento em uma nuvem pública pode gerar o vazamento desses dados.
- Ameaças de aprendizado de máquina: processos de aprendizado de máquina são vulneráveis a ataques de segurança, com ataques podendo influenciar o seu treinamento e diminuindo a performance e confiabilidade do sistema.

Ambientes virtualizados são vantajosos para a construção de sistemas de controle industriais (ICS, *Industrial Control Systems*), que necessitam manter o sistema se comportando dentro de valores esperados. ICS podem ser virtualizados e executados na nuvem, executando funções virtualizadas para efetuar o controle das operações de comunicação e segurança. Gêmeos digitais podem ser utilizados em sistemas como esse para servirem de mesas de teste para experimentação, geração de cenários de otimização, análise forense para identificar causas de mal funcionamento ou incidentes de segurança, tudo para garantir uma interoperabilidade segura nos sistemas controlados efetuando uma avaliação dinâmica dos eventos críticos do sistema [37], [62]. Dai et al [63] propõe o uso de gêmeos digitais para criar um sistema de defesa baseado em defesa através de mímica criando uma estrutura dinâmica de redundância, o que significa que os resultados de processamento são comparados vindos de múltiplas entidades diferentes, sendo fácil identificar um membro que não está operando da forma correta, ou seja, que foi atacado e modificado.

Com o acoplamento direto entre o mundo real e digital, mecanismos mais eficientes para defesa dos sistemas são necessários. Liu [64] sugere a introdução de um plano de segurança, dedicado ao suporte nativo de segurança na rede, composto por três partes: (1) Operação e Manutenção de segurança inteligente, (2) motor inteligente de políticas de segurança e (3) uma biblioteca habilitadora de segurança. (1) conduz as operações de operação e manutenção (O&M) baseados em inteligência artificial e informações obtidas no plano de dados; (2) o motor dinamicamente ajusta e melhora as políticas de segurança da rede e os mecanismos de segurança utilizados pelos componentes da rede; e (3) planeja as funções de segurança que serão requisitadas pela rede. Pelo uso de inteligência artificial, redes terão segurança nativa implementada, através de interação e colaboração entre as entidades conectadas, criando imunidade proativa de ameaças de forma ubíqua na rede [64].

Devido aos limites computacionais, capacidades de segurança, armazenamento e recursos de rede da computação em borda, além da grande quantidade de pontos de acesso, é difícil de ela estar equipada com medidas de proteção efetivas. Sun et al [65] apresenta uma arquitetura de proteção especificamente para computação em borda, com recomendações para camada de aplicação, dados, rede, nodo e de recurso, criando um sistema de proteção e mecanismos de segurança que levam em consideração a flexibilidade de gêmeos digitais na borda.

Provisionamento de políticas de acesso em setores industriais requer pré-análise, definição de papéis, permissões e regras para acessar recursos e efetuar ações, considerando restrições precisas e consistentes, o que é uma atividade complexa de administrar, devido a decentralização dos sistemas. Logo, administração de sistemas de autorização devem se tornar automatizados, com miníma intervenção manual, com o uso de inteligência artificial para duas atividades: **Alteração Automática de Política**, criando compreensão de acessos anteriores e o seu efeito no sistema, para refinar regras existentes. Mineração de dados e algoritmos de classificação podem ser usados para identificar falhas em especificações de políticas; e **Aprendizado de Regras**, onde algoritmos são treinados partindo de dados e inferindo novas políticas, através de aprendizado por reforço [66].

Para ambientes industriais, Krishnan et al [67] propõe um

TABLE I
CLASSIFICAÇÃO DE TRABALHOS

Categorias	Trabalhos	Total
5G, 6G e Sistemas *Wireless*	[2], [4], [13], [15]–[20], [22]–[37], [39]–[42], [47], [64], [66]	27
Detecção de Anomalias	[10], [11]	2
Escalonamento de VNF	[20]	1
Gêmeos Digitais para Redes	[1], [3]–[10], [12], [21], [30], [31], [46], [52], [59]	16
Internet das Coisas	[2], [14], [25], [31], [39], [43]–[52], [62], [67]	17
P4	[19], [21]	2
Redes Ópticas	[58]–[60]	3
Redes Veiculares	[54]–[57]	4
Segurança	[4], [37], [61]–[67]	9

sistema de detecção de intrusos utilizando a descrição de uso do fabricante (*manufacturer usage description*, MUD) dos dispositivos para melhorar o monitoramento de redes, explorando se dispositivos IoT apresentam padrões previsíveis, que podem ser definidos formal e sucintamente, utilizando gêmeos digitais e redes definidas por software para aumentar a segurança de ambientes industriais.

IV. CONCLUSÃO

A lista de artigos lidos gerou um total de 9 categorias de interesse, com base nos assuntos que cada trabalha apresentava, e podem ser visualizados na tabela I. Alguns trabalhos se encaixaram em mais de uma categoria, então a classificação não é exclusiva. Para quem tem interesse em um assunto específico, esta pesquisa serve como um breve resumo de contribuições e relações dos três assuntos-chave (SDN, NFV e gêmeos digitais) com as categorias originadas.

Uma classificação dentro de cada categoria seria uma próxima etapa dentro desse trabalho, para diferenciar as contribuições dentro de cada disciplina por cada um dos trabalhos.

Gêmeos digitais (e seus muitos outros nomes e subclassificações) representam a integração do mundo físico com o mundo digital. Através da obtenção de grandes quantidades de dados e com estudo de padrões por aprendizado de máquina, torna possível sistemas realmente autônomos, que aprendem e se tornam mais eficientes sem necessidade de intervenção humana. Sua inserção tecnológica torna possível a ubiquidade da computação, com dispositivos conectados em todos os lugares, transparente ao ser humano. Para a indústria, ela representa mais estabilidade, eficiência e redução de custos em todas esferas, tornando possível níveis de precisão sem limites humanos.

No contexto de redes de computadores, gêmeos digitais, implementados com o auxílio de SDN e NFV, auxiliam na solução de problemas na rede, predição do estado futuro da rede e melhora na sua confiabilidade por eliminar riscos conhecidos antes de acontecerem, além de poderem ser ferramentas úteis para automação e aumento de segurança de sistemas.

REFERENCES

[1] M. Alja'Afreh, "A qoe model for digital twin systems in the era of the tactile internet," Ph.D. dissertation, Université d'Ottawa/University of Ottawa, 2021.

[2] C. Nguyen and D. Hoang, "Software-defined virtual sensors for provisioning iot services on demand," in *2020 5th International Conference on Computer and Communication Systems (ICCCS)*. IEEE, 2020, pp. 796–802.

[3] C. Zhou, H. Yang, X. Duan, D. Lopez, A. Pastor, Q. Wu, M. Boucadair, and C. Jacquenet, "Digital Twin Network: Concepts and Reference Architecture," Internet Engineering Task Force, Internet-Draft draft-zhou-nmrg-digitaltwin-network-concepts-07, Mar. 2022, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-zhou-nmrg-digitaltwin-network-concepts-07

[4] L. U. Khan, Z. Han, W. Saad, E. Hossain, M. Guizani, and C. S. Hong, "Digital twin of wireless systems: Overview, taxonomy, challenges, and opportunities," *arXiv preprint arXiv:2202.02559*, 2022.

[5] J. Jagannath, K. Ramezanpour, and A. Jagannath, "Digital twin virtualization with machine learning for iot and beyond 5g networks: Research directions for security and optimal control," *arXiv preprint arXiv:2204.01950*, 2022.

[6] M. Ferriol-Galmés, J. Suárez-Varela, J. Paillisse, X. Shi, S. Xiao, X. Cheng, P. Barlet-Ros, and A. Cabellos-Aparicio, "Building a digital twin for network optimization using graph neural networks," *Available at SSRN 3995236*, 2021.

[7] S. Vakaruk, A. Mozo, A. Pastor, and D. R. López, "A digital twin network for security training in 5g industrial environments," in *2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI)*. IEEE, 2021, pp. 395–398.

[8] C. Güemes-Palau, P. Almasan, S. Xiao, X. Cheng, X. Shi, P. Barlet-Ros, and A. Cabellos-Aparicio, "Accelerating deep reinforcement learning for digital twin network optimization with evolutionary strategies," *arXiv preprint arXiv:2202.00360*, 2022.

[9] P. Almasan, M. Ferriol-Galmés, J. Paillisse, J. Suárez-Varela, D. Perino, D. López, A. A. P. Perales, P. Harvey, L. Ciavaglia, L. Wong *et al.*, "Digital twin network: Opportunities and challenges," *arXiv preprint arXiv:2201.01144*, 2022.

[10] W. Wang, L. Tang, C. Wang, and Q. Chen, "Real-time analysis of multiple root causes for anomalies assisted by digital twin in nfv environment," *IEEE Transactions on Network and Service Management*, 2022.

[11] W. Wang, Q. Chen, T. Liu, and L. Tang, "Digital-twin assisted root cause analysis of anomalies in nfv environment," in *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[12] X. Sun, C. Zhou, X. Duan, and T. Sun, "A digital twin network solution for end-to-end network service level agreement (sla) assurance," *Digital Twin*, vol. 1, no. 5, p. 5, 2021.

[13] L. U. Khan, W. Saad, D. Niyato, Z. Han, and C. S. Hong, "Digital-twin-enabled 6g: Vision, architectural trends, and future directions," *arXiv preprint arXiv:2102.12169*, 2021.

[14] M. Kherbache, M. Maimour, and E. Rondeau, "When digital twin meets network softwarization in the industrial iot: Real-time requirements case study," *Sensors*, vol. 21, no. 24, p. 8194, 2021.

[15] S. Schneider, M. Peuster, K. Hannemann, D. Behnke, M. Müller, P.-B. Bøk, and H. Karl, ""producing cloud-native": Smart manufacturing use cases on kubernetes," in *2019 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. IEEE, 2019, pp. 1–2.

[16] F. Granelli, R. Capraro, M. Lorandi, and P. Casari, "Evaluating a digital twin of an iot resource slice: an emulation study using the eliot platform," *IEEE Networking Letters*, vol. 3, no. 3, pp. 147–151, 2021.

[17] F. Naeem, G. Kaddoum, and M. Tariq, "Digital twin-empowered network slicing in b5g networks: Experience-driven approach," in *2021 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2021, pp. 1–5.

[18] A. Fellan, C. Schellenberger, M. Zimmermann, and H. D. Schotten, "Enabling communication technologies for automated unmanned vehicles in industry 4.0," in *2018 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2018, pp. 171–176.

[19] C.-Y. Chang, T. G. Ruiz, F. Paolucci, M. A. Jiménez, J. Sacido, C. Papagianni, F. Ubaldi, D. Scano, M. Gharbaoui, A. Giorgetti *et al.*, "Performance isolation for network slices in industry 4.0: The 5growth approach," *IEEE Access*, vol. 9, pp. 166 990–167 003, 2021.

[20] E. Zeydan, J. Mangues-Bafalluy, J. Baranda, R. Martínez, and L. Vettori, "A multi-criteria decision making approach for scaling and placement of virtual network functions," *Journal of Network and Systems Management*, vol. 30, no. 2, pp. 1–36, 2022.

[21] R. Larsson, "Creating digital twin distributed networks using switches with programmable data plane," 2021.

[22] H. Wang, Y. Wu, G. Min, and W. Miao, "A graph neural network-based digital twin for network slicing management," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 2, pp. 1367–1376, 2020.

[23] D. de Vleeschauwer, J. Baranda, J. Mangues-Bafalluy, C. F. Chiasserini, M. Malinverno, C. Puligheddu, L. Magoula, J. Martín-Pérez, S. Barmpounakis, K. Kondepu *et al.*, "5growth data-driven ai-based scaling," in *2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*. IEEE, 2021, pp. 383–388.

[24] B. Han, W. Jiang, M. A. Habibi, and H. D. Schotten, "An abstracted survey on 6g: Drivers, requirements, efforts, and enablers," *arXiv preprint arXiv:2101.01062*, 2021.

[25] M. Peuster, S. Schneider, D. Behnke, M. Müller, P.-B. Bøk, and H. Karl, "Prototyping and demonstrating 5g verticals: the smart manufacturing case," in *2019 IEEE Conference on Network Softwarization (NetSoft)*. IEEE, 2019, pp. 236–238.

[26] G. Liu, N. Li, J. Deng, Y. Wang, J. Sun, and Y. Huang, "6g mobile network architecture-solids: Driving forces, features, and functional topology," *Engineering*, 2021.

[27] J. T. Penttinen, "On 6g visions and requirements," *Journal of ICT Standardization*, pp. 311–326, 2021.

[28] M. Tariq, F. Naeem, and H. V. Poor, "Toward experience-driven traffic management and orchestration in digital-twin-enabled 6g networks," *arXiv preprint arXiv:2201.04259*, 2022.

[29] S. Yrjölä *et al.*, "Decentralized 6g business models," *Proceedings of the 6G Wirel. Summit, Levi, Finland*, pp. 5–7, 2019.

[30] H. Ahmadi, A. Nag, Z. Khan, K. Sayrafian, and S. Rahadrja, "Networked twins and twins of networks: an overview on the relationship between digital twins and 6g," *arXiv preprint arXiv:2108.05781*, 2021.

[31] X. Shen, J. Gao, W. Wu, M. Li, C. Zhou, and W. Zhuang, "Holistic network virtualization and pervasive network intelligence for 6g," *IEEE Communications Surveys & Tutorials*, 2021.

[32] S. Kumar, "6g mobile communication networks: Key services and enabling technologies," *Journal of ICT Standardization*, pp. 1–10, 2022.

[33] D. Stock, M. Schneider, and T. Bauernhansl, "Towards asset administration shell-based resource virtualization in 5g architecture-enabled cyber-physical production systems," *Procedia CIRP*, vol. 104, pp. 945–950, 2021.

[34] N. S. Kumar, U. Kaur, T. Anuradha, S. Majji, S. R. Karanam, and R. G. Deshmukh, "5g network virtualization for the remote driving enhancement," in *2021 4th International Conference on Computing and Communications Technologies (ICCCT)*. IEEE, 2021, pp. 458–463.

[35] E.-K. Hong, I. Lee, B. Shim, Y.-C. Ko, S.-H. Kim, S. Pack, K. Lee, S. Kim, J.-H. Kim, Y. Shin *et al.*, "6g r&d vision: Requirements and candidate technologies," *Journal of Communications and Networks*, NA.

[36] J. Cheng, Y. Yang, X. Zou, and Y. Zuo, "5g in manufacturing: a literature review and future research," *The International Journal of Advanced Manufacturing Technology*, pp. 1–23, 2022.

[37] A. Narayan, C. Krueger, A. Goering, D. Babazadeh, M.-C. Harre, B. Wortelen, A. Luedtke, and S. Lehnhoff, "Towards future scada systems for ict-reliant energy systems," in *International ETG-Congress 2019; ETG Symposium*. VDE, 2019, pp. 1–7.

[38] A. Filali, A. Abouaomar, S. Cherkaoui, A. Kobbane, and M. Guizani, "Multi-access edge computing: A survey," *IEEE Access*, vol. 8, pp. 197 017–197 046, 2020.

[39] M. Groshev, C. Guimarães, A. De La Oliva, and R. Gazda, "Dissecting the impact of information and communication technologies on digital twins as a service," *IEEE Access*, vol. 9, pp. 102 862–102 876, 2021.

[40] S. Schneider, M. Peuster, D. Behnke, M. Müller, P.-B. Bøk, and H. Karl, "Putting 5g into production: Realizing a smart manufacturing vertical scenario," in *2019 European Conference on Networks and Communications (EuCNC)*. IEEE, 2019, pp. 305–309.

[41] D. Roy, A. S. Rao, T. Alpcan, G. Das, and M. Palaniswami, "Achieving ai-enabled robust end-to-end quality of experience over radio access networks," *arXiv preprint arXiv:2201.05184*, 2022.

[42] J. M. Taylor and H. R. Sharif, "Leveraging digital twins to enhance performance of iot in disadvantaged networks," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2020, pp. 1303–1308.

[43] R. Minerva, G. M. Lee, and N. Crespi, "Digital twin in the iot context: a survey on technical features, scenarios, and architectural models," *Proceedings of the IEEE*, vol. 108, no. 10, pp. 1785–1824, 2020.

[44] S. Yun, J.-h. Park, H.-s. Kim, and W.-T. Kim, "Importance-aware sdn control mechanism for real-time data distribution services," in *2018 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2018, pp. 1113–1118.

[45] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-iiotset: A new comprehensive realistic cyber security dataset of iot and iiot applications for centralized and federated learning," *IEEE Access*, vol. 10, pp. 40 281–40 306, 2022.

[46] T. Yang, J. Chen, and N. Zhang, "Ai-empowered maritime internet of things: a parallel-network-driven approach," *IEEE Network*, vol. 34, no. 5, pp. 54–59, 2020.

[47] S. Zeb, A. Mahmood, S. A. Hassan, M. J. Piran, M. Gidlund, and M. Guizani, "Industrial digital twins at the nexus of nextg wireless networks and computational intelligence: A survey," *Journal of Network and Computer Applications*, p. 103309, 2022.

[48] A. K. Ghosh, A. S. Ullah, R. Teti, and A. Kubo, "Developing sensor signal-based digital twins for intelligent machine tools," *Journal of Industrial Information Integration*, vol. 24, p. 100242, 2021.

[49] P. Bellavista, C. Giannelli, M. Mamei, M. Mendula, and M. Picone, "Application-driven network-aware digital twin management in industrial edge environments," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7791–7801, 2021.

[50] I. A. Tsokalo, D. Kuss, I. Kharabet, F. H. Fitzek, and M. Reisslein, "Remote robot control with human-in-the-loop over long distances using digital twins," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[51] J. Santa, J. Ortiz, P. J. Fernandez, M. Luis, C. Gomes, J. Oliveira, D. Gomes, R. Sanchez-Iborra, S. Sargento, and A. F. Skarmeta, "Migrate: Mobile device virtualisation through state transfer," *IEEE Access*, vol. 8, pp. 25 848–25 862, 2020.

[52] H. Chahed and A. J. Kassler, "Software-defined time sensitive networks configuration and management," in *2021 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. IEEE, 2021, pp. 124–128.

[53] H. Kim, H. Shin, H.-s. Kim, and W.-T. Kim, "Vr-cpes: A novel cyber-physical education systems for interactive vr services based on a mobile platform," *Mobile Information Systems*, vol. 2018, 2018.

[54] L. Zhao, G. Han, Z. Li, and L. Shu, "Intelligent digital twin-based software-defined vehicular networks," *IEEE Network*, vol. 34, no. 5, pp. 178–184, 2020.

[55] S. Almeaibed, S. Al-Rubaye, A. Tsourdos, and N. P. Avdelidis, "Digital twin analysis to promote safety and security in autonomous vehicles," *IEEE Communications Standards Magazine*, vol. 5, no. 1, pp. 40–46, 2021.

[56] Y. Dai and Y. Zhang, "Adaptive digital twin for vehicular edge computing and networks," *Journal of Communications and Information Networks*, vol. 7, no. 1, pp. 48–59, 2022.

[57] A. A. Zaid, B. E. Y. Belmekki, and M.-S. Alouini, "Technological trends and key communication enablers for evtols," *arXiv preprint arXiv:2110.08830*, 2021.

[58] D. S. N. A. B. Pg, S. S. Newaz, F. H. Rahman, T.-W. Au, N. S. Nafi, R. K. Patchmuthu, F. Al-Hazemi *et al.*, "Digital-twin-assisted software-defined pon: A cognition-driven framework for energy conservation," in *2021 31st International Telecommunication Networks and Applications Conference (ITNAC)*. IEEE, 2021, pp. 166–177.

[59] M. G. Alabarce and P. P. Mariño, "Optical network design and analysis tools: A test of time," *Optical Switching and Networking*, vol. 44, p. 100651, 2022.

[60] D. Kilper, J. Yu, and S. Santaniello, "Optical networking in smart city and wireless future networks platforms," in *2021 European Conference on Optical Communication (ECOC)*. IEEE, 2021, pp. 1–4.

[61] E. Karaarslan and M. Babiker, "Digital twin security threats and countermeasures: An introduction," in *2021 International Conference on Information Security and Cryptology (ISCTURKEY)*. IEEE, 2021, pp. 7–11.

[62] A. F. Murillo and S. Rueda, "Access control policies for network function virtualization environments in industrial control systems," in *2020 4th Conference on Cloud and Internet of Things (CIoT)*. IEEE, 2020, pp. 17–24.

[63] W. Dai, S. Li, L. Lu, Y. Ye, F. Meng, and D. Zhang, "Research on application of mimic defense in industrial control system security," in *2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA)*, vol. 2. IEEE, 2021, pp. 573–577.

[64] G. Liu, Y. Huang, N. Li, J. Dong, J. Jin, Q. Wang, and N. Li, "Vision, requirements and network architecture of 6g mobile network beyond 2030," *China Communications*, vol. 17, no. 9, pp. 92–104, 2020.

[65] Y. Sun, X. Xu, R. Qiang, and Q. Yuan, "Research on security management and control of power grid digital twin based on edge computing," in *2021 2nd International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT)*. IEEE, 2021, pp. 606–610.

[66] J. Lopez, J. E. Rubio, and C. Alcaraz, "Digital twins for intelligent authorization in the b5g-enabled smart grid," *IEEE Wireless Communications*, vol. 28, no. 2, pp. 48–55, 2021.

[67] P. Krishnan, K. Jain, R. Buyya, P. Vijayakumar, A. Nayyar, M. Bilal, and H. Song, "Mud-based behavioral profiling security framework for software-defined iot networks," *IEEE Internet of Things Journal*, 2021.

APPENDIX D — AVALIAÇÃO DE UMA ARQUITETURA BASEADA EM MQTT E KAFKA PARA GÊMEOS DIGITAIS (PORTUGUESE)

The following appended document is the final works for the CMP258 course, Desenvolvimento de Aplicações Big Data, Cloud e Fog/EDGE, written in Portuguese. It presents early results in prototyping the Kafka/MQTT cluster implementation.

The increasing adoption of digital twins to model and simulate real systems is a sign of progression in the implementation of the Internet of Things in various industry sectors. Due to the progressive increase in the amount of data transmitted and processed, the use of distributed systems in the cloud becomes the viable alternative for implementing digital twins. However, dispatching computation to the cloud introduces the problem of added latency, both in the communication of data and in the processing time of data, which may be overloaded with data, causing devices at the edge to have to wait for the result to process and the propagation time back across the network. To solve this problem, we propose the use of intermediate systems to the network, which will manage the data communication, through an MQTT communication based on clusters and pre-processed by Apache Kafka, so that the system can automatically scale its components to maintain the whole operation of the platform, without delays and data losses for the application, according to its computational demand.

FRANCISCO PAIVA KNEBEL

Avaliação de uma arquitetura baseada em MQTT e Kafka para gêmeos digitais

Orientador: Prof. Dr. Juliano Araújo Wickboldt

Porto Alegre
2021

RESUMO

A crescente adoção do uso de gêmeos digitais para modelar e simular sistemas reais é um sinal de progressão na implementação da Internet das Coisas em diversos setores da indústria. Devido ao aumento progressivo da quantidade de dados que são transmitidos e processados, o uso de sistemas distribuídos na nuvem se torna a alternativa viável para implementação de gêmeos digitais. Entretanto, o despacho de computação para a nuvem introduz o problema de adição de latência, tanto na comunicação dos dados, que devem trafegar por múltiplos pontos até chegar ao seu destino, quanto no tempo de processamento dos mesmos, que podem estar sobrecarregados com dados, fazendo com que os dispositivos na borda terão que esperar o resultado ser computado, além do tempo de propagação de volta por toda a rede. Para resolver esse problema, propomos a utilização de sistemas intermediários à rede, que irão gerenciar a comunicação dos dados, através de uma comunicação MQTT baseada em *clusters* e pré-processada pelo Apache Kafka, de forma que o sistema consiga de forma automática escalar os seus componentes para manter o funcionamento pleno da plataforma, sem atrasos e perdas de dados para a aplicação, de acordo com a sua demanda computacional.

Palavras-chave: Gêmeo digital. internet das coisas. computacao em nuvem. MQTT. Kafka.

Evaluation of an architecture based on MQTT and Kafka for digital twins

ABSTRACT

The increasing adoption of digital twins to model and simulate real systems is a sign of progression in the implementation of the Internet of Things in various industry sectors. Due to the progressive increase in the amount of data transmitted and processed, the use of distributed systems in the cloud becomes the viable alternative for implementing digital twins. However, dispatching computation to the cloud introduces the problem of added latency, both in the communication of data and in the processing time of data, which may be overloaded with data, causing devices at the edge to have to wait for the result to process and the propagation time back across the network. To solve this problem, we propose the use of intermediate systems to the network, which will manage the data communication, through an MQTT communication based on clusters and pre-processed by Apache Kafka, so that the system can automatically scale its components to maintain the whole operation of the platform, without delays and data losses for the application, according to its computational demand.

Keywords: Digital twin, internet of things, cloud computing, MQTT, Kafka.

SUMÁRIO

1 INTRODUÇÃO

O conceito de Gêmeo Digital (*Digital Twin*), introduzido em 2002 como um modelo de um sistema formado por dois espaços espelhados e interconectados, um real e um virtual, onde cada um reflete o seu parceiro, ao mesmo tempo que se comunicam em tempo real (GRIEVES, 2016). Em outras palavras, a parte virtual de um gêmeo digital é um programa de computador, com capacidade de gerar simulações sobre o funcionamento do sistema real e como ele irá ser afetado por dados recebidos via o uso de sensores, capturando suas características e sendo capaz de revelar problemas do sistema, de forma proativa e agindo de forma autônoma (BOSCHERT; ROSEN, 2016).

A expansão de dispositivos sensores de baixo custo, devido à Internet das Coisas (*Internet of Things*, IoT) é um dos principais fatores motores ao desenvolvimento de gêmeos digitais, sendo uma solução puramente digital e de fácil implementação, não precisando uma duplicação física do sistema, e sim apenas uma definição dos requisitos que o compõem e alimentação de dados para seus algoritmos.

Com essa definição, a pesquisa em gêmeos digitais é um assunto atual de extrema relevância e de muito desenvolvimento tanto pela indústria quanto pela academia.

1.1 Projeto PeTWIN

Objetivado pela digitalização do processo produtivo em gêmeos digitais que permitem simular todas as condições de produção de uma planta de petróleo, o projeto PeTWIN - Gêmeos Digitais para Otimização e Gerenciamento de Produção é uma parceria entre o Grupo de Pesquisa Sistemas de Computação para E&P de Petróleo do INF-UFRGS e o Sirius Lab da Universidade de Oslo, possível através da Financiadora de Estudos e Projetos (FINEP) e o conselho norueguês de pesquisa (RCN), cofinanciados pelas empresas Petrobras, Shell Total, CNN e CNOC, que compõem o consórcio de Libra no Brasil e Shell Equinor norueguesas.

O projeto PeTWIN tem como foco oferecer uma visão científica e as melhores práticas para a implementação de gêmeos digitais sustentáveis, utilizáveis e de fácil manutenção para o gerenciamento de campo. Simplificando, um gêmeo digital é um repositório de dados associado a um simulador que monitora a operação de uma instalação de produção de petróleo permitindo análise de dados on-line e avaliação preditiva. O PeTWIN abordará os desafios da ciência da computação apresentados por gêmeos digi-

tais por meio da aplicação e desenvolvimento de técnicas de captura e processamento de dados físicos em tempo real, a modelagem conceitual destes dados sob uma visão integrada e a análise de dados. Embora gêmeos digitais sejam uma tecnologia conhecida para equipamentos industriais, a aplicação de um gêmeo digital para um ativo em produção de petróleo é bastante nova e inovadora em termos de análise integrada de dados.

1.2 Open Digital Twin

Motivado por esse projeto, foram desenvolvidos alguns trabalhos prévios ao seu início, de forma a estudar e entender melhor o estado da arte na pesquisa de gêmeos digitais. Um problema levantado na pesquisa literária sobre o assunto foi a dificuldade de encontrar implementações abertas e de fácil reprodutibilidade. Devido à isso, inicialmente tivemos o foco de construção de uma arquitetura aberta para gêmeos digitais, construída partindo de uma coleção de microsserviços e hospedada em uma organização no GitHub chamada Open Digital Twin[1], onde é desenvolvido o *software* e armazenado dados de experimentação. Esse estudo gerou a escrita de três trabalhos na área de gêmeos digitais, antes do seu vínculo com o projeto PeTWIN.

Durante o semestre letivo de 2020/1, como Aluno Especial na disciplina "CMP158 - Real Time Systems" do PPGC, foi feito um trabalho conectando os temas de gêmeos digitais com sistemas de tempo real, onde foi observado que para que gêmeos digitais com requisitos de tempo real dependem não apenas no resultado lógico de sua computação, mas também de suas limitações temporais, e, para lidar com a grande quantidade de dados que devem ser armazenados e analisados, gêmeos digitais dependem de arquiteturas baseadas na nuvem, o que gera um fator adicional de latência na computação efetuada, que pode ser atenuado com o uso de computação em névoa, reduzindo os tempos de resposta para respeitar seus requisitos de tempo (KNEBEL; WICKBOLDT; FREITAS, 2020).

Outro trabalho paralelo foi desenvolvido na arquitetura de gêmeos digitais do Open Digital Twin, analisando os limites de implantação e comunicação via uso do protocolo MQTT (*Message Queuing Telemetry Transport*), avaliando o uso do tipo de comunicação *Publisher-Subscriber* no contexto de gêmeos digitais, observando limites do *broker* que gerencia a entrega de mensagens e examinando o comportando do mesmo em situações centralizadas e distribuídas, com mais e menos recursos atribuídos (TREVISAN;

[1] Disponível em `https://github.com/Open-Digital-Twin`.

KNEBEL; WICKBOLDT, 2020).

E por fim, um terceiro trabalho conectando toda a pesquisa feita até então, efetuando um levantamento bibliográfico sobre gêmeos digitais, contextualização, seus casos de uso e plataformas estabelecidas, apresentando a arquitetura desenvolvida pelo Open Digital Twin e suas motivações para ser um sistema aberto e baseado em microsserviços, de forma a ser flexível e com complexidade reduzida para integração com aplicativos de terceiros. O estudo obteve redução de até 64% no tempo de transmissão média de mensagens para o gêmeo digital ao implantar em nós de computação em névoa, em comparação com uma abordagem exclusivamente feita em nuvem (KNEBEL, 2020).

A organização Open Digital Twin continua sendo um espaço utilizado para desenvolvimento de protótipos para a arquitetura de gêmeos digitais, de forma agnóstica ao projeto, sem possuir vínculo direto, mas sendo um espaço aberto para experimentação e desenvolvimento da arquitetura protótipo para gêmeos digitais.

2 OBJETIVOS E DEFINIÇÕES

Evoluindo as pesquisas previamente efetuadas pelo grupo, nos trabalhos anteriormente citados, avaliamos que ainda existem muitos desafios para a implementação plena de gêmeos digitais, de uma forma escalável e sustentável.

Latência é um problema de muita preocupação com a comunicação IoT, especialmente no contexto de aplicações industriais, e com a maior adaptação e implementação, serviços de baixa latência se tornarão cada vez mais importantes (FERRARI *et al.*, 2017). Com o crescimento da demanda e a criação de comunicação em larga escala, a quantidade de tráfego na rede pode ser um limitante para a resultante inovação tecnológica vinculada à gêmeos digitais, onde fatores como escalabilidade do sistema precisam ser consideradas na sua implementação.

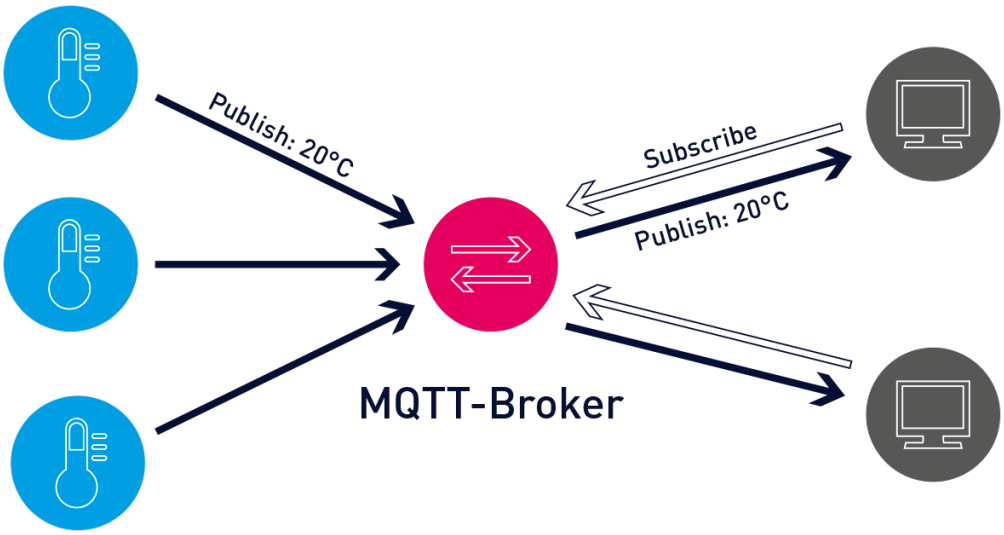
O uso de MQTT para resolver esses problemas no contexto de IoT e gêmeos digitais é amplamente documentado (GRGIĆ; ŠPEH; HEĐI, 2016; ATMOKO; RIANTINI; HASIN, 2017; JACOBY; USLÄNDER, 2020), sendo que seu objetivo é o de ser um protocolo leve de troca de mensagens, que pode ser executado por dispositivos com pouco poder computacional e conectados em redes de baixa confiabilidade, amplamente utilizado e padronizado para uso por dispositivos IoT. Em uma arquitetura MQTT, há dois tipos de sistemas: clientes e *brokers*. Um *broker* é um servidor intermediário que clientes se comunicam, recebendo e transmitindo dados, sem conexão direta entre os clientes.

Apesar de ser um protocolo dito leve, mensagens podem ser configuradas para maior garantia de entrega, como ao definir a Qualidade do Serviço (*Quality of Service*, QoS). Em situações onde a rede é confiável mas limitada, o uso de QoS 0, que apenas envia os dados sem receber confirmação, pode ser a melhor escolha, mas o QoS 1 e 2 permitem maior garantia sobre a entrega de mensagem, com o cliente e o *broker* se comunicando para ter certeza que a mensagem chegou no seu destino.

A Figura 2.1 apresenta um exemplo dessa comunicação, com o envio de dados por sensores de temperatura com um *Publish* (Publicar, enviar os dados para o *broker*) e o recebimento do mesmo por outros dispositivos com um *Subscribe* (Inscrição, pedir para o *broker* repassar as informações desejadas). Mensagens no contexto de MQTT são publicadas e identificadas dentro do *broker* em "tópicos", que representam de onde esse dado veio, de qual dispositivo dentro da arquitetura desenvolvida, e clientes escolhem quais tópicos se inscrever.

Para resolver isso de forma que não haja prejuízo na comunicação, através da

Figura 2.1 – Exemplo de comunicação *Publish/Subscribe*, usando MQTT.



Fonte: `https://www.paessler.com/it-explained/mqtt`

perda de mensagens, o funcionamento do *broker* MQTT em modo *cluster*, ou seja, múltiplos *brokers* ligados entre si e trabalhando em conjunto, permite obter melhor desempenho (maior poder de processamento), disponibilidade (sistema resistente a falhas, redundante), e efetuando balanceamento de carga (distribuição equilibrada do processamento).

Após o recebimento dos dados pelo *cluster*, o problema de latência ainda não foi resolvido, pois ainda há problemas em relação ao tempo de processamento da aplicação. No caso de recebimento de muitos dados, esperado para o aumento de complexidade introduzido pelos gêmeos digitais, gera a necessidade de um sistema igualmente capaz de processar eles valores, de acordo com suas especificações, ou seja, uma infraestrutura robusta, com capacidade para o armazenamento e o processamento de grandes quantidades de dados em tempo de execução é requerida.

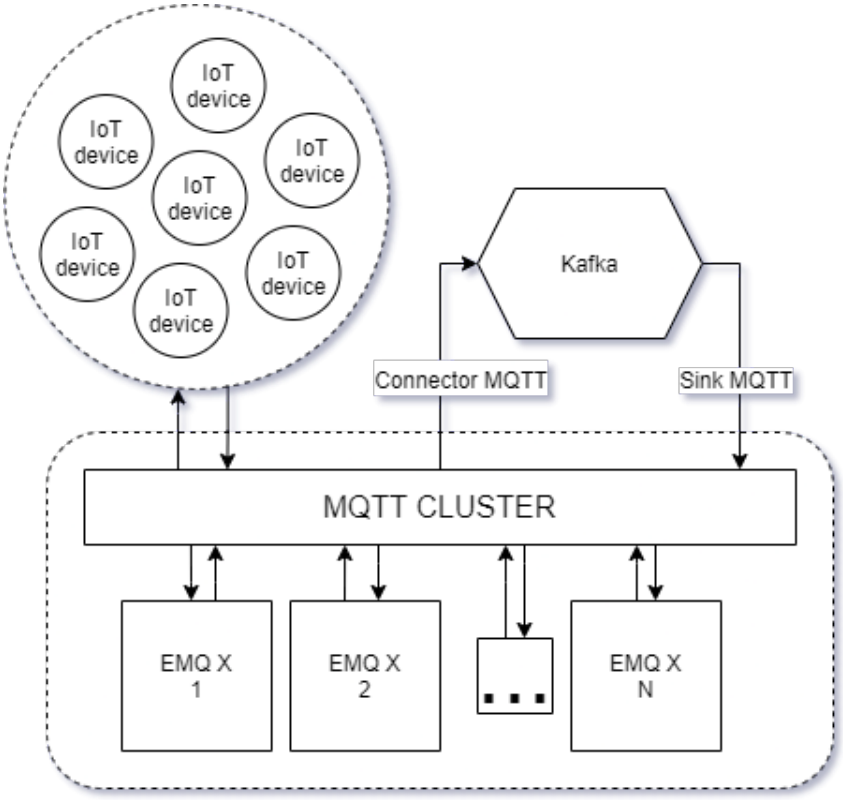
Uma parte crucial na implementação de um gêmeo digital é o tratamento para os dados adquiridos. Após serem coletados, os dados recebidos devem ser divididos das seguintes formas, para processamento:

- **Em tempo real**, o que significa que há uma entrada de dados contínua e o processamento deve seguir regras estritas de tempo, com pouco espaço de variação.
- **Quase em tempo-real**, ou um tempo real *leve*, onde as regras de tempo são mais flexíveis e mais relacionadas à percepção humana, sendo que a qualidade do resultado degrada com o aumento do tempo de computação.
- **Em lote**, de forma a processar volumes grandes de dados coletados num período de

tempo logo, onde não há preocupação sobre o tempo de computação, e sim sobre o resultado de todo o conjunto para aquela operação.

Na arquitetura proposta, o Apache Kafka é introduzido para mitigar esse problema, de forma à operar como um sistema de tempo real leve. Possuindo capacidade de lidar com um alto número de entradas de dado em tempo real, o Kafka pode ser distribuído, particionado e replicado (D'SILVA *et al.*, 2017), permitindo uma computação em uma escala maior de comunicação de variados tipos e cargas de dados, de uma maneira confiável, permitindo o uso de aplicações de tempo real para grandes volumes de dados (SOUSA *et al.*, 2018).

Figura 2.2 – Arquitetura simplificada do sistema proposto.



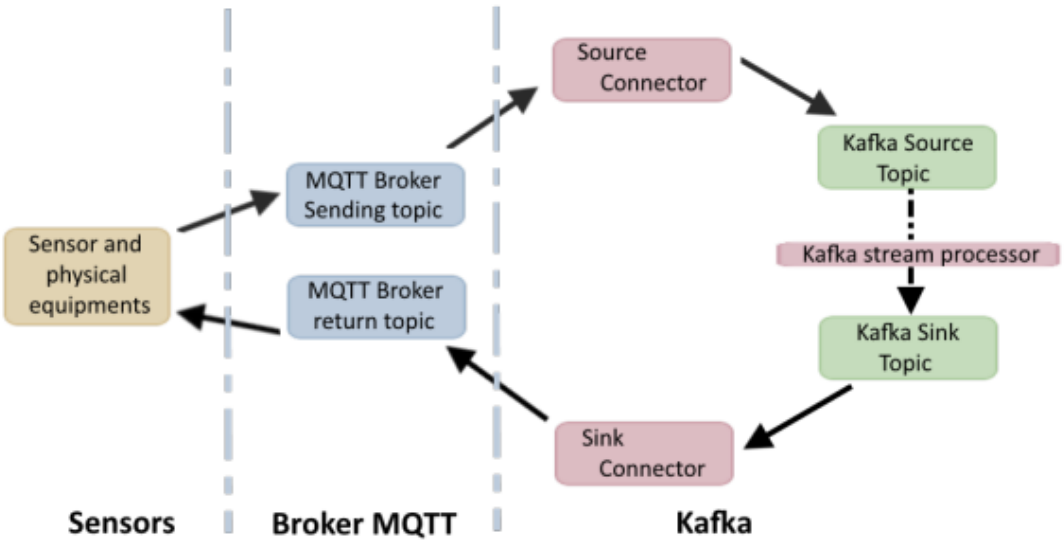
Fonte: Os Autores

Como objetivo, o trabalho a ser desenvolvido visa a avaliação do desempenho da comunicação entre dispositivos IoT e *Cloud* através do uso de comunicação MQTT, para transmissão de mensagens, e Apache Kafka, para processamento de *streams*, utilizando clusterização de *brokers* num contexto de computação de névoa e borda. Além da implementação do sistema, será feito uma análise de funcionalidades dessas ferramentas, como clusterização e particionamento de dados, com o foco no teste de integração de MQTT e

Apache Kafka para comunicação e processamento IoT.

A Figura 2.2 representa uma visão simplificada da arquitetura proposta. Utilizando o EMQ X como *broker* MQTT, escolhido por ser uma implementação capaz de altas taxas de transferência (KOZIOLEK; GRÜNER; RÜCKERT, 2020), que possuirá um conjunto de *brokers* MQTT funcionando em modo de *cluster*, será feito uma análise sobre o sistema durante o recebimento de mensagens de múltiplos dispositivos IoT. O Apache Kafka, através de um conector *Source* de MQTT conectado ao *cluster* MQTT, receberá esses dados para processamento, retornando resultados do seu processamento para o *cluster* MQTT, através de um conector *Sink* de MQTT, uma saída de dados do Kafka de volta para o *cluster*. A Figura 2.3 detalha o fluxo de dados dentro dessa estrutura, com os sensores gerando dados para o *broker* MQTT, o *Source* trazendo a informação para o Kafka, onde ele é processado, e transmitido de volta para o *broker* via o *Sink*, onde os dispositivos de origem podem obter a computação feita pelo processo do gêmeo digital.

Figura 2.3 – Fluxo de dados do sistema proposto.



Fonte: Os Autores

3 IMPLEMENTAÇÃO

Com a arquitetura apresentada em mente, e com as motivações apresentadas no capítulo 2 para escolha das tecnologias, foi feito a implementação do protótipo, utilizando o EMQ X como *broker* MQTT e o Apache Kafka para o processamento de *streams*. Toda implementação está disponível e acessível de forma aberta, na organização do Open Digital Twin no GitHub [1].

3.1 Confluent

Uma peça não definida para o protótipo é como será feito a conexão MQTT-Kafka. Devido à existência da API Kafka Connect, o conceito de registrar *Sources* e *Sinks* é uma forma comum para essa conexão, um componente do próprio Apache Kafka para executar a integração de *streaming* entre o Kafka e outros sistemas, como bancos de dados, serviços de nuvem e um *broker* MQTT.

A Confluent apresenta uma implementação pronta para uso dentro do seu ecossistema do Kafka [2], permitindo a definição de novos conectores com o uso apenas de um arquivo JSON, configurando as opções desejadas. As Figuras 3.1a e 3.1b apresentam as definições utilizadas para os conectores utilizados, definindo o fluxo MQTT-Kafka-MQTT.

Figura 3.1 – Definições dos conectores utilizados no Kafka Connect.
(a) Conector *Sink*. (b) Conector *Source*.

Fonte: Os Autores

[1] Repositório do projeto disponível em `https://github.com/Open-Digital-Twin/test-kafka-emqx-cluster`.

[2] Mais informações em `https://www.confluent.io/product/confluent-connectors/`.

3.2 Consumidor

Depois do conector trazer a informação para um tópico do Kafka, uma aplicação *Subscriber* precisa estar conectada ao nodo para ler esse tópico. Devido à isso, três exemplos diferentes de *Subscribers* foram implementados, utilizando o node-rdkafka [3], utilizando APIs diferentes do Kafka.

A primeira implementação utiliza a Stream API, representado na Figura 3.2, e possui a lógica mais simples de operação, criando uma *stream* de dados, e ao receber, executa uma função definida pela aplicação.

Figura 3.2 – Consumidor usando a API *Streaming*.

```
export class ConsumerStreaming extends Consumer {
  type = 'CONSUMER_STREAMING'
  stream

  constructor(settings, producer) {
    super(settings, producer)

    this.init(this.type)
  }

  consume() {
    this.stream = Kafka.KafkaConsumer.createReadStream(
      this.globalConfig,
      {},
      {
        topics: [this.topic],
      }
    )

    this.stream
      .on('data', (message) => this.handleData(message))
      .on('error', function (err) {
        console.error('Error in our kafka consume stream')
        console.error(err)
      })
  }
}

```

Fonte: Os Autores

A segunda e a terceira, apresentados nas Figuras 3.3a e 3.3b, utilizam a API padrão para *Subscriber*, de duas formas: *flowing* e *non-flowing*. O modo *flowing* mantém um loop infinito na execução do programa, trazendo todas as mensagens, até detectar que a conexão foi encerrada. O modo *non-flowing* lê N mensagens de forma manual, explicitamente.

[3] Implementação disponível em `https://github.com/blizzard/node-rdkafka`.

Figura 3.3 – Consumidores usando a API *Standard*.

(a) Consumidor *Standard Flowing*.

```
export class ConsumerStandardFlowing extends Consumer {
  type = 'CONSUMER_STANDARD_FLOWING'
  consumer

  constructor(settings, producer) {
    super(settings, producer)

    this.init(this.type)
  }

  consume() {
    this.consumer.connect()

    this.consumer
      .on('ready', () => {
        console.log(`Consuming records from topic "${this.topic}"`)
        this.consumer.subscribe([this.topic])
        this.consumer.consume()
      })
      .on('data', (message) => this.handleData(message))
  }
}
```

(b) Consumidor *Standard Non-flowing*.

```
export class ConsumerStandardNonFlowing extends Consumer {
  type = 'CONSUMER_STANDARD_NON_FLOWING'
  consumer

  constructor(settings, producer) {
    super(settings, producer)

    this.init(this.type)
  }

  consume() {
    this.consumer.connect()

    this.consumer
      .on('ready', () => {
        console.log(`Consuming records from topic "${this.topic}"`)
        this.consumer.subscribe([this.topic])

        setInterval(() => {
          this.consumer.consume(100)
        }, 100)
      })
      .on('data', (message) => this.handleData(message))
  }
}
```

Fonte: Os Autores

As três implementações de consumidores utilizaram uma implementação comum em forma de classe, estendendo a classe da Figura 3.4a, que inicializa o componente e define como os dados recebidos serão processados, de acordo com os algoritmos de teste da Figura 3.4b. Esses algoritmos efetuam operações com os dados em níveis de complexidade diferentes, de forma à analisar como a ferramenta de *streaming* vai ser capaz de processar dados de formas diferentes.

Figura 3.4 – Consumidor-base e algoritmos.

(a) Classe Consumidora base.

```
class Consumer {
    type;
    topic;
    kafkaBrokerList;
    seen = 0;

    globalConfig;

    constructor(settings, producer) {
        this.topic = settings.sourceTopic;
        this.kafkaBrokerList = settings.kafkaBrokerList;

        this.producer = producer;

        this.globalConfig = {
            'group.id': 'kafka',
            'metadata.broker.list': this.kafkaBrokerList,
        };
    }

    handleData({ value, size, topic, offset, partition, key, timestamp }) {
        console.log(`Consumed record with key ${key} and value ${value} of partition ${partition} @
offset ${offset}. Updated total count to ${++this.seen}`);

        // const product = calculate_pi(this.seen).toString();
        // const product = calculate_exponent(value.toString().split(' ')[0], this.seen).toString();
        // const product = calculate_fibonacci(this.seen).toString();
        const product = calculate_fibonacci_memory(this.seen).toString();

        if (this.producer) {
            this.producer.produce(product);
        }
    }

    init(type) {
        switch (type) {
            case 'CONSUMER_STANDARD_FLOWING':
            case 'CONSUMER_STANDARD_NON_FLOWING':
                this.consumer = new Kafka.KafkaConsumer(this.globalConfig, {});

                process.on('SIGINT', () => {
                    console.log('\nDisconnecting consumer ...');
                    this.consumer.disconnect();
                });
            case 'CONSUMER_STREAMING':
                break;
            default:
                console.log('Wrong consumer type.');
        }
    }
}
```

(b) Algoritmos de processamento.

```
function calculate_pi(digits) {
    let i = 1n;
    let x = 3n * (10n ** (BigInt(digits) + 20n));
    let pi = x;

    while (x > 0) {
        for (let j = 0; j < 100; ++j) {
            x = x * i / ((i + 1n) * 4n);
            pi += x / (i + 2n);
            i += 2n;
        }
    }

    return '3.' + (pi / (10n ** 20n)).toString(10).substr(1);
}

// O(log N)
function calculate_exponent(n, k) {
    const n_big = BigInt(value.toString().split(' ')[0])
    const k_big = BigInt(this.seen)

    const product = (k_big ** n_big).toString(10);
}

// O(2^N)
function calculate_fibonacci(n) {
    if (n <= 1) {
        return n;
    }

    return calculate_fibonacci(n - 2) + calculate_fibonacci(n - 1);
}


// O(N-2)
const fibonacci_memory = {
    0: BigInt(0),
    1: BigInt(1)
};

function calculate_fibonacci_memory(n) {
    if (n in fibonacci_memory) {
        return fibonacci_memory[n];
    }

    fibonacci_memory[n] = calculate_fibonacci_memory(n-2) + calculate_fibonacci_memory(n-1);
    return fibonacci_memory[n];
}
```

Fonte: Os Autores

3.3 Produtor

Da mesma forma que o Consumidor, a construção do Produtor também permite o uso das diferentes APIs do Kafka. Para essa implementação, duas formas diferentes foram utilizadas: uma utilizando a API padrão, representado na Figura 3.5a, e outra utilizando a API de *Streaming*, representando na Figura 3.5b.

Figura 3.5 – Produtores usando as APIs *Standard* e *Streaming*.
(a) Produtor *Standard*.

```
export class ProducerStandard extends Producer {
  type = 'PRODUCER_STANDARD'
  producer

  constructor(settings) {
    super(settings)

    this.producer = new Kafka.Producer({
      'metadata.broker.list': settings.kafkaBrokerList,
    })

    this.producer.connect()

    this.producer.on('event.error', function (err) {
      console.error('Error from producer')
      console.error(err)
    })

    this.producer.setPollInterval(100)
  }

  produce(value) {
    try {
      console.log('Producing via standard mode.')

      this.producer.produce(
        this.topic,
        null,
        Buffer.from(value),
        null,
        Date.now()
      )
    } catch (err) {
      console.error('A problem occurred when producing')
      console.error(err)
    }
  }
}
```

(b) Produtor *Streaming*.

```
export class ProducerStreaming extends Producer {
  type = 'PRODUCER_STREAMING'
  stream

  constructor(settings) {
    super(settings)

    stream = Kafka.Producer.createWriteStream(
      {
        'metadata.broker.list': settings.kafkaBrokerList,
      },
      {},
      {
        topic: this.topic,
      }
    )
  }

  produce(value) {
    console.log('Producing via streaming mode.')

    // Writes a message to the stream
    const queuedSuccess = this.stream.write(Buffer.from(value))

    if (queuedSuccess) {
      console.log('Queued message.')
    } else {
      // Note that this only tells us if the stream's queue is full,
      // it does NOT tell us if the message got to Kafka!  See below...
      console.log('Too many messages in queue.')
    }

    this.stream.on('error', function (err) {
      console.error('Error in our kafka stream')
      console.error(err)
    })
  }
}
```

Fonte: Os Autores

3.4 Cliente MQTT

Para executar o papel de gerador de dados, utilizamos uma implementação aberta feita pelo grupo denominada *dt-client-bytes* [4], capaz de facilmente gerar dados aleatórios, com tamanho e frequência de envio customizados, para *brokers* MQTT. Essa solução está presente nos repositórios do Open Digital Twin no GitHub. Essa implementação é facilmente customizável e integrável com o ambiente utilizado, de forma que podemos emular e escalar qualquer quantidade de clientes apenas alterando o arquivo de configuração do ambiente. O receptor dos dados também é uma implementação própria [5] com o objetivo de captar o máximo de dados possíveis do *broker* MQTT em que está inscrito.

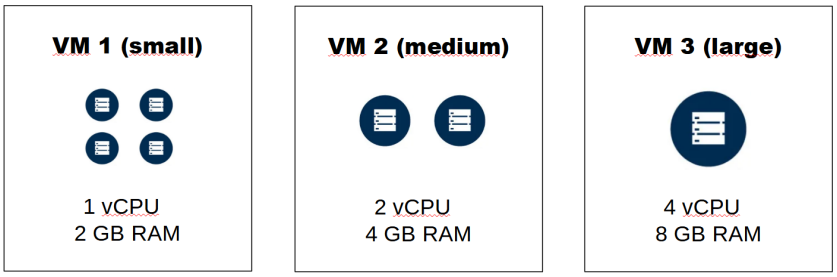
[4] Disponível em `https://github.com/Open-Digital-Twin/dt-client-bytes`.
[5] Disponível em `https://github.com/Open-Digital-Twin/dt-instance`.

4 EXPERIMENTAÇÃO

4.1 Sobre o *broker* MQTT

Como o EMQ X, a implementação do MQTT escolhida, tem como algumas de suas funcionalidades o funcionamento em modo de *cluster* e a priorização de mensagens, separamos a experimentação do *broker* na arquitetura em duas formas: teste de **escalabilidade** e de **tempo real**, onde foram testados ambas funcionalidades. Na Figura 4.1, estão apresentados três tipos diferentes de máquinas virtuais que foram utilizadas para a experimentação. A ideia principal é manter a mesma quantidade de recursos disponíveis para a aplicação, mas testando se a sua distribuição afetaria o funcionamento da aplicação. Para os histogramas apresentados, a frequência apresentada deve ser multiplicada por um fator de 10 para obter a quantidade de mensagens real enviadas, devido à taxa de amostragem.

Figura 4.1 – Tipos de máquinas utilizados nos experimentos.



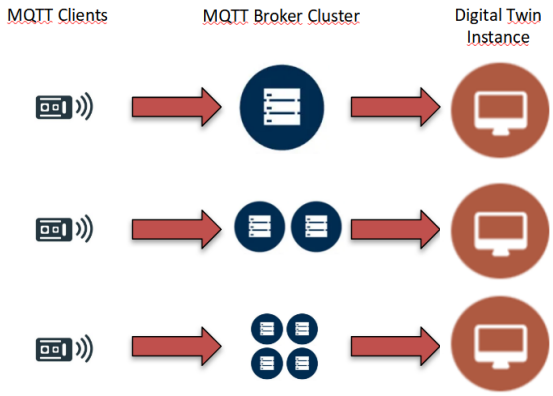
Fonte: Os Autores

4.1.1 Escalabilidade

Para o teste de escalabilidade, três cenários foram construídos, como descrito na Figura 4.2. O primeiro cenário utiliza apenas um *broker*, rodando na máquina grande, com mais recursos. O segundo cenário reduz a quantidade de recursos por máquina, mas começa a utilizar o modo de *cluster*, conectando dois *brokers*, cada um executando em máquinas médias, com metade dos recursos da máquina grande. Já o terceiro experimento segue a mesma lógica, mas conectando quatro *brokers* em máquinas pequenas, com metade dos recursos da máquina média. Dessa forma, o hardware utilizado para cada experimento é o mesmo, mas operando sobre modos de execução diferentes.

Em todos os três cenários, foram introduzidos 40 clientes MQTT simultâneos, ge-

Figura 4.2 – Definição do uso das máquinas nos experimentos de escalabilidade do EMQ X.

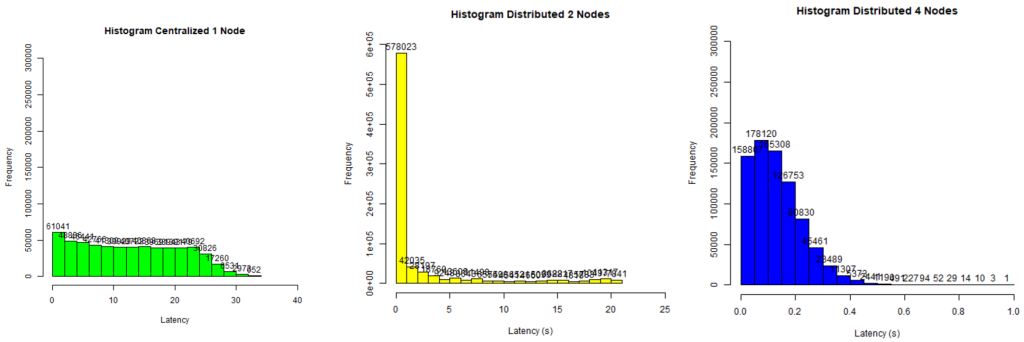


Fonte: Os Autores

rando dados para o *broker*, sendo lidos por apenas um inscrito, enviando dados em uma taxa fixa de 200 mensagens por segundo, com tamanho de 64 bytes cada, em 4 tópicos diferentes. O total de mensagens enviadas em cada cenário foi de 8 milhões de mensagens, o que representa 200.000 mensagens sendo enviadas por cada cliente, de forma simultânea, utilizando a mesma rede, em um prazo de 1000 segundos, o que consideramos um cenário representativo de uma grande rede de sensores, para o contexto de gêmeos digitais. A quantidade de clientes ajuda a balancear a rede, de forma a distribuir a quantidade de mensagens sendo enviadas para cada *broker*, garantindo pelo balanceamento de dados que a computação sobre o envio seja dividido entre os *brokers*.

Figura 4.3 – Histogramas das mensagens para os três experimentos.

(a) Resultado 1 *broker* G. (b) Resultado 2 *broker* M. (c) Produtor 4 *broker* P.



Fonte: Os Autores

A execução dos três cenários geraram várias conclusões. A Figura 4.3 apresenta histogramas dos tempos das mensagens, do cliente gerador até chegar ao destino. No cenário de 1 *broker* G, houve a geração de filas, o que causou atraso na propagação dos dados, pois apenas um *broker* não conseguiu lidar com a quantidade de mensagens sendo recebidas. Também foi observado uma perda considerável de mensagens, com 2.213.410

mensagens perdidas, do total de 8.000.000 enviadas, observando-se uma latência máxima de 34 segundos. O efeito de enfileiramento também ocorreu sobre o cenário de 2 *broker* M, mas menos atenuado, com grande parte das mensagens com latências inferiores à faixa dos 5 segundos, e 72.25% das mensagens foram entregues em menos de 1 segundo. Entretanto, os dois *brokers* distribuídos foram suficientes para lidar com o fluxo de mensagens recebidas e não necessitar descarte, gerando perda zero sobre o total de mensagens. A latência entretanto ainda esteve presente, devido ao enfileiramento, com o máximo observado na faixa dos 22 segundos. Já o cenário de 4 *brokers* P não apresentou a geração de filas, evidenciado na Figura 4.3c, onde a totalidade das mensagens foram recebidas num intervalo inferior à 1 segundo. A distribuição de *brokers* foi suficiente para receber a quantidade de mensagens, sem perdas e com enfileiramento mínimo.

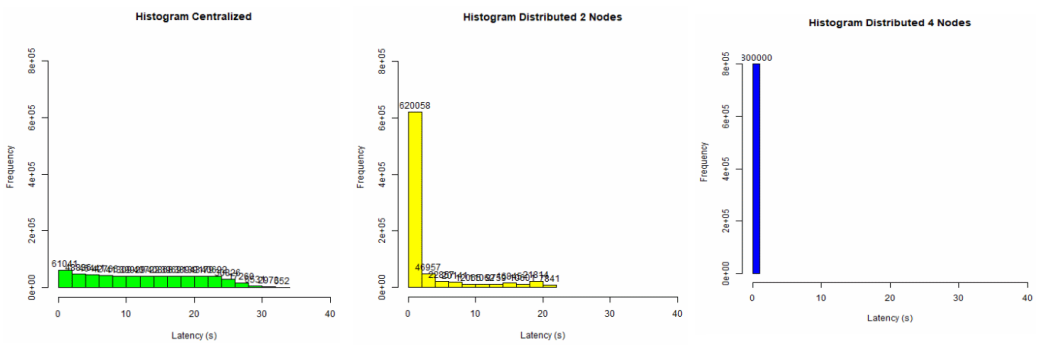
A Tabela 4.1 apresenta um resultado resumido dos três experimentos. Na Figura 4.4, os histogramas do tempo de mensagem para os três experimentos estão apresentados na mesma escala de tempo, ficando evidente a atenuação do comportamento de enfileiramento e a diminuição da latência inserida com a distribuição de *brokers* dos experimentos.

Tabela 4.1 – Resultados agregados sobre o experimento de *clusters* no EMQ X.

Cenário	Perda de mensagens?	Taxa de mensagens (msg/s)	Latência Máxima (s)
1 *Broker* G	Sim	3810,72	33,80
2 *Broker* M	Não	5504,30	20,08
4 *Broker* P	Não	4683,80	0,91

Fonte: Os Autores

Figura 4.4 – Histogramas das mensagens para os três experimentos, na mesma escala.
(a) Resultado 1 *broker* G. (b) Resultado 2 *broker* M. (c) Produtor 4 *broker* P.

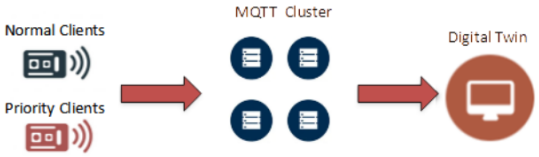


Fonte: Os Autores

4.1.2 Tempo Real

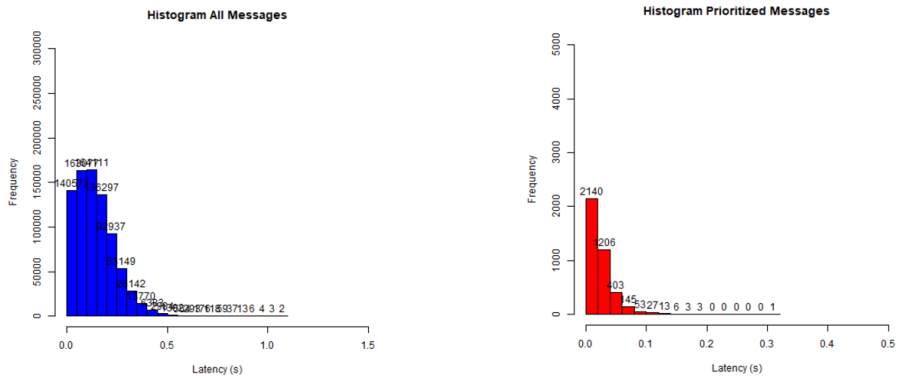
Para o experimento focado na priorização de mensagens, foi utilizado o cenário de melhor desempenho da experimentação de escalabilidade, mantendo o uso de 4 *brokers* pequenos, demonstrado na Figura 4.6. Para a geração de dados, foram utilizados 40 clientes geradores de dados de baixa prioridade, com as mesmas taxas dos cenários anteriores. Entretanto, para testar a priorização, foram introduzidos 4 clientes de alta prioridade. Esses clientes possuem uma taxa inferior de geração de mensagens, enviado 10 mensagens por segundo, ao contrário da taxa de 200 dos clientes de baixa prioridade, mas que o *broker* foi configurado para dar prioridade, para garantir a entrega o mais rápido possível do que tiver maior prioridade.

Figura 4.5 – Definição do experimento de priorização de mensagens.



Fonte: Os Autores

Figura 4.6 – Histogramas com os resultados do experimento de priorização de mensagens.
(a) Histograma de todas mensagens. (b) Histograma de mensagens priorizadas.



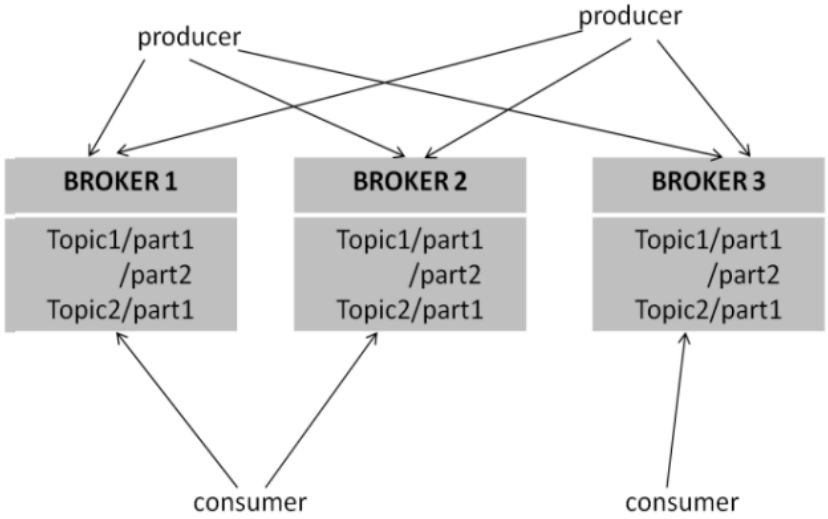
Fonte: Os Autores

Os resultados obtidos estão evidenciados na Figura 4.6, com 4.6a apresentando o histograma de todas mensagens enviadas e 4.6b apenas o histograma das mensagens priorizadas. Podem ser observados que os resultados obtidos da execução do experimento foram mantidos (a inexistência do enfileiramento e a redução de latência) e que 98.6% das mensagens de alta prioridade obtiveram uma latência observada menor do que 1 segundo.

4.2 Sobre o Apache Kafka

O Apache Kafka também funciona num modelo *publish/subscribe* baseado em tópicos. Para o Kafka, tópicos são categorias de mensagens, que podem ser armazenados em uma ou mais partições, que são os depósitos físicos dentro de um *cluster* Kafka. Partições de um tópico podem estar distribuídos sobre o *cluster*, criando mais pontos de entrada para os dados e permitindo leituras e escritas em paralelo, além de poderem ser replicados, aumentando a tolerância a falhas (WU; SHANG; WOLTER, 2019), como exemplificado na Figura 4.7.

Figura 4.7 – Exemplo de separação de tópicos em um *cluster* Kafka.



Fonte: (THEIN, 2014)

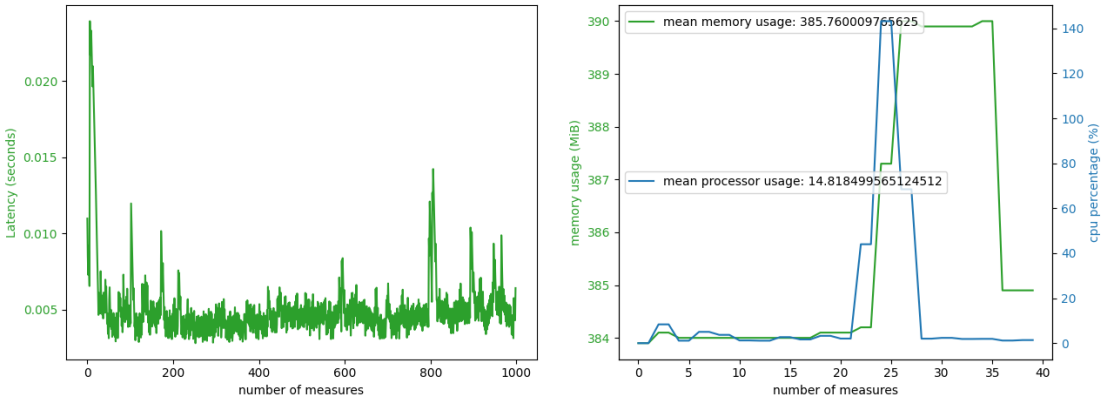
Da mesma forma que tínhamos a preocupação com o recebimento de mensagens no *cluster* MQTT, temos a mesma preocupação de escalabilidade e com o tempo de execução com o *cluster* Kafka. Um experimento foi efetuados para testar o tempo de transmissão das mensagens sobre o Kafka, enviando um total de 1000 mensagens, com cada uma sendo enviada a cada 1ms. Na 4.8, os resultados do tempo de transmissão da mensagem e um gráfico sobre o uso de memória e de processador são apresentados.

Analisando os resultados obtidos, algumas conclusões podem ser retiradas. Primeiramente, o tempo de transmissão médio entre a comunicação Kafka e MQTT é bem distinta, com média de 0.005s de latência observada no Kafka. Em relação ao uso de recursos, foram alocados um valor fixo na faixa dos 400 megabytes de memória ao longo de todo o tempo mensurado, com o Apache Kafka tirando proveito dos dois vCPUs disponíveis na máquina utilizada, atingindo 140% de uso. Mais variações das experimentações

estão sendo efetuadas, com a inclusão de *clusters* e particionamento, para fontes de dados maiores, mas esse experimento provou o rápido processamento de mensagens prometido pelo Kafka, necessário para lidar com o *streaming* de dados desejado.

Figura 4.8 – Resultados sobre o experimento com o Kafka.

(a) Tempo de transmissão. (b) Uso de recursos ao longo do experimento.



Fonte: Os Autores

5 CONCLUSÃO

O trabalho efetuado ainda possui muitos pontos a serem expandidos, que o serão feitos para a dissertação de mestrado. A avaliação do processo inteiro da mensagem (*round-trip*) ainda precisa ser efetuado, e experimentação efetuada em situações mais representativas de sistemas existentes na realidade.

Para a experimentação sobre MQTT, mais discussões estão sendo efetuadas e uma avaliação sobre o comportamento de clusterização e priorização de mensagens é um trabalho que está sendo efetuado, visando submissão nos próximos meses. Um trabalho futuro sendo discutido é o de auto-regulação do *broker*, analisando o funcionamento do *cluster* para automaticamente escalar o mesmo de acordo com a quantidade de dados sendo transmitida.

Em relação ao Apache Kafka, com o sistema funcional implementado, agora será explorado mais de suas funcionalidades e modelado sua forma de uso e integração com a arquitetura de gêmeo digital.

REFERÊNCIAS

ATMOKO, R.A.; RIANTINI, R.; HASIN, M.K. IoT real time data acquisition using MQTT protocol. *In* IOP PUBLISHING, 1. JOURNAL of Physics: Conference Series. [*S. l.: s. n.*], 2017. v. 853, p. 012003.

BOSCHERT, Stefan; ROSEN, Roland. Digital Twin—The Simulation Aspect. *In*. MECHATRONIC Futures. [*S. l.*]: Springer, 2016. p. 59–74.

D'SILVA, Godson Michael *et al.* Real-time processing of IoT events with historic data using Apache Kafka and Apache Spark with dashing framework. *In*. 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT). [*S. l.: s. n.*], 2017. p. 1804–1809. DOI: 10.1109/RTEICT.2017.8256910.

FERRARI, Paolo *et al.* Evaluation of communication latency in industrial IoT applications. *In* IEEE. 2017 IEEE International Workshop on Measurement and Networking (M&N). [*S. l.: s. n.*], 2017. p. 1–6.

GRGIĆ, Krešimir; ŠPEH, Ivan; HEĐI, Ivan. A web-based IoT solution for monitoring data using MQTT protocol. *In* IEEE. 2016 international conference on smart systems and technologies (SST). [*S. l.: s. n.*], 2016. p. 249–253.

GRIEVES, Michael. **Origins of the Digital Twin Concept**. [*S. l.*], 2016.

JACOBY, Michael; USLÄNDER, Thomas. Digital Twin and Internet of Things—Current Standards Landscape. **Applied Sciences**, Multidisciplinary Digital Publishing Institute, v. 10, n. 18, p. 6519, 2020.

KNEBEL, Francisco Paiva. An open Digital Twin framework based on microservices in the cloud, 2020.

KNEBEL, Francisco Paiva; WICKBOLDT, Juliano Araujo; FREITAS, Edison Pignaton de. A Cloud-Fog Computing Architecture for Real-Time Digital Twins. **arXiv preprint arXiv:2012.06118**, 2020. Disponível em: https://arxiv.org/abs/2012.06118.

KOZIOLEK, Heiko; GRÜNER, Sten; RÜCKERT, Julius. A Comparison of MQTT Brokers for Distributed IoT Edge Computing. *In* SPRINGER. EUROPEAN Conference on Software Architecture. [*S. l.: s. n.*], 2020. p. 352–368.

SOUSA, Adriano B de *et al.* Uma plataforma de IoT para integração de dispositivos baseada em nuvem com Apache Kafka. *In* SBC. ANAIS Estendidos do XXXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. Porto Alegre,

RS, Brasil: SBC, 2018. DOI: 10.5753/sbrc_estendido.2018.14638. Disponível em: https://sol.sbc.org.br/index.php/sbrc_estendido/article/view/14638.

THEIN, Khin Me Me. Apache Kafka: Next Generation Distributed Messaging System. **International Journal of Scientific Engineering and Technology Research**, v. 3, n. 47, p. 9478–9483, 2014.

TREVISAN, Rafael; KNEBEL, Francisco; WICKBOLDT, Juliano. Uma Avaliação do uso de MQTT para a Implementação de Digital Twins. *In*. ANAIS da XVIII Escola Regional de Redes de Computadores. Evento Online: SBC, 2020. p. 41–47. DOI: 10.5753/errc.2020.15187. Disponível em: https://sol.sbc.org.br/index.php/errc/article/view/15187.

WU, Han; SHANG, Zhihao; WOLTER, Katinka. Performance Prediction for the Apache Kafka Messaging System. *In*. 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS). [*S. l.: s. n.*], 2019. p. 154–161. DOI: 10.1109/HPCC/SmartCity/DSS.2019.00036.