Universidade de Aveiro
2023

MARIANA
PINTO

COMPUTAÇÃO CONTÍNUA EM AMBIENTE DE REDE FEDERADA

COMPUTING CONTINUUM FOR A FEDERATED NETWORK FABRIC

Universidade de Aveiro
2023

MARIANA
PINTO

COMPUTAÇÃO CONTÍNUA EM AMBIENTE DE REDE FEDERADA

COMPUTING CONTINUUM FOR A FEDERATED NETWORK FABRIC

“*It is the unknown we fear when we look upon death and darkness, nothing more.*”

— Albus Dumbledore

Universidade de Aveiro
2023

MARIANA
PINTO

COMPUTAÇÃO CONTÍNUA EM AMBIENTE DE REDE FEDERADA

COMPUTING CONTINUUM FOR A FEDERATED NETWORK FABRIC

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor Diogo Gomes, Professor associado do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

Dedico este trabalho à Mariana de 18 anos que não sabia o que lhe esperava.
Conseguimos!

o júri / the jury

presidente / president	Professor Doutor António Manuel Duarte Nogueira Professor Auxiliar da Universidade de Aveiro
vogais / examiners committee	Professor Doutor Bruno Miguel de Oliveira e Sousa Professor Auxiliar da Universidade de Coimbra - Faculdade de Ciências e Tecnologia
	Professor Doutor Diogo Nuno Pereira Gomes Professor Auxiliar, da Universidade de Aveiro

agradecimentos / acknowledgements

Se há 7 anos atrás me tivessem dito que estaria aqui hoje, iria rir-me. Foram dos anos mais duros, mas também mais desafiantes da minha vida, tanto a nível pessoal, de saúde, monetário e até no trabalho. Cresci, chorei, ri, realizei sonhos, perdi sonhos, mas hoje sinto-me muito feliz.

Aproveito este espaço para agradecer a toda a gente que me acompanhou nesta jornada. Ao meu orientador, Professor Diogo Gomes, muito obrigada por ter aceite o meu desafio. Obrigada por todas as reuniões, pela paciência, pela insistência e não me deixar desistir. A sua ajuda e os seus ensinamentos irão comigo para a vida, como orientador mas também como dos melhores professores do DETI.

À Ubiwhere, em especial ao André Duarte, agradeço toda a ajuda que me deram e de todas as oportunidades que me foram surgindo e aparecendo, tanto neste projeto, como trabalhadora a full-time. Aos meus colegas de trabalho, obrigada por todos os sorrisos e encorajamento.

Aos meus colegas de faculdade e amigos por estarem sempre lá para mim, principalmente o saxofone solos que nunca me deixou sozinha.

À minha família que me apoiou sempre todos estes anos e sem a qual isto não seria possível. Pai, mãe, tia, tio, mana, cunhado, Bea, Tiago, Nanda, Toni, Simão, Tweety, Diego, estão sempre no meu coração.

À família que eu escolhi, o meu Cacho por me distraírem sempre que eu precisava e terem tido a paciência de me aturar estes últimos 13 anos. Obrigada por tudo.

Ao meu Snoopy, por toda a companhia que me fez e nunca me ter deixado sozinha nas longas horas de escrita deste documento.

Finalmente, à pessoa mais importante da minha vida, por ter sido a minha âncora durante todo o meu percurso, por nunca ter saído do meu lado mesmo quando eu não merecia, por estar a construir uma vida comigo e por ter sido o único que sabe de todas as dificuldades que passamos: Bruno, amo-te. Não mudaria uma vírgula na minha vida, só para poder ter o privilégio de nos ver a conquistar tudo o que estamos a conquistar.

Palavras Chave

Nuvem, Borda, Internet das Coisas, Rede, Federada, Ambiente, Computação Contínua, Latência, Compatibilidade a Multi Acesso de Computação de Borda

Resumo

Esta dissertação explora no conceito de computação contínua para uma rede federada e embarca na investigação do seu papel fundamental na superação da divisão digital que frequentemente separa diferentes ambientes computacionais, aprofundando a integração dos recursos de computação na cloud, edge e em data centers. Consegue oferecer uma perspetiva compreensiva de como estes elementos podem coexistir e colaborar harmoniosamente. O principal objtivo desta pesquisa envolve a avaliação de uma potencial rede federada para otimizar alocação de recursos, aprimorar gestão de dados e aumentar a eficiencia geral de um sistema. Para além disso, o estudo aprofunda as possibilidades da rede federada de permitir a distribuição dinâmica de carga de trabalho e a orquestração em ambientes computacionais heterogêneos. Através de análises meticulosas e casos de estudos ilustrativos, esta dissertação elucida as vantagens e desafios relacionados ao Continuum Computacional para uma Estrutura de Rede Federada. Ao abordar preocupações cruciais relacionadas à escalabilidade, segurança e interoperabilidade, ela fornece insights importantes sobre o futuro da computação em rede, com foco na conquista de uma paisagem digital unificada, eficiente e abrangente.

Keywords

Cloud , Edge, Internet of things, Network, Federated, Fabric, Computing Continuum, Latency, Multi Access Edge Computing Compatibility

Abstract

This thesis delves into the concept of a "Computing Continuum for a Federated Network Fabric" and embarks on an exploration of its pivotal role in bridging the digital divide that often partitions distinct computational environments. It delves deeply into the integration of edge, cloud and data center computing resources, offering a perspective on how these elements can coexist and collaborate harmoniously. The primary objectives of this research revolve around evaluating the potential of federated network fabrics to optimize resource allocation, enhance data management, and elevate overall system efficiency. Furthermore, the study delves into the possibilities of federated network fabrics enabling dynamic workload distribution and orchestration within heterogeneous computing environments. Through analysis and illustrative case studies, this thesis aspires to elucidate the advantages and challenges linked to the Computing Continuum for a Federated Network Fabric. By addressing pivotal concerns pertaining to scalability, security, and interoperability, it furnishes invaluable insights into the future of networked computing, with a focal point on attaining a unified, efficient, and all-encompassing digital landscape.

Contents

List of Figures

List of Tables

Glossary

ACID	Atomicity, Consistency, Isolation, and Durability
AI	Artificial Inteligent
AMQP	Advanced Message Queuing Protoco
API	Application Programming Interface
CATR	Chinese Academy of Telecommunications Research
CDN	Content Delivery Network
CPU	Central Processing Unit
EC-RA	Edge Computing Reference Architecture
ECC	Edge Computing Consortium
ETSI	European Telecommunications Standards Institute
FTP	File Transfer Protocol
GECA	Global Edge Computing Architecture
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IIA	Industrial Internet Alliance
IIC-RA	Industrial Internet Consortium Reference Architecture
IMAP	nternet Message Access Protoco
IoT	Internet of Things
IP	Internet Protocol
IPAM	IP Address Management
ISEO	Intel Smart Edge Open
IT	Information Technology
MEC	Multi-Access Edge Computing
MIIT	Ministry of Industry and Information Technology
MQTT	Message Queuing Telemetry Transport
NFV	Network Functions Virtualization
POP3	Post Office Protocol version 3
QoS	Quality of Service
REST	Representational State Transfer
RTP	Real-time Transport Protocol
SD-WAN	Software-Defined Wide Area Network
SIP	Session Initiation Protoco
SDN	Software-Defined Networking
SLA	Service Level Agreement
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell)
TCP	Transmission Control Protoco
UDP	User Datagram Protocol
UI	User Interface

CHAPTER 1

Introduction

The COVID -19 pandemic and associated lockdowns have led to fundamental changes in the way we live and work, accelerating the need for people to become closer virtually and giving rise to new models of communication. In a world of constant change, it is essential that these mechanisms are responsive and (almost) always available, hence the need to study new, robust and secure solutions for communication between machines, in order to make the right decisions when analyzing the data coming into the system. The increasing prevalence of Internet of Things (IoT) devices and the data collected by these devices and systems, have led the industry to rethink how data is processed. Processing data in real-time at the edge of the network allows companies to gain more immediate insights from connected devices and systems. By being able to analyse data directly at the source, businesses can make decisions and take actions based on the most current data at any given point in time, leading to more efficient and assertive decisions.

As online networks started to grow and more new vulnerabilities in the broader Information Technology (IT) architectures emerged, many companies started researching new ways of being able to work remotely to increase the automation of operational and repetitive tasks, in order to reduce production and infrastructure costs. The amount of data moving in (and out) of an enterprise network today makes old data compute models obsolete: rather than continuing to rely on a traditional data center compute model, the industry has embraced the concept of a computing continuum - placing the right computing resources at the optimal processing points in the system - from cloud data centers to edge systems and endpoint devices. [1]

Due to the lack of standardised solutions comes the need and urgency to develop more architectures that provide a secure and fast environment for data exchange between machines and facilities. The impact of industrial IoT is quickly disrupting the balance between on-premises and cloud data centres. Think of modern production facilities that require a high concentration of computing power on site: A robotic arm should not wait a millisecond longer than necessary to start the next operation, and if it fails, it must inform the system as soon as possible. Thus the need for constant communication between machines, ideally with as little delay as possible, in order not to jeopardise tasks and expected results, and to manage applications in such a way that if one machine fails, no other fails and the system can recover itself. In an ideal world, all machines communicate with each other without latency, interruptions, data leakage, connection problems or intrusions, making 100% correct decisions with the data they got. However, it is not always the case: security breaches, delays in information exchange, network failures, power outages, and outdated systems - can put physical infrastructure, health, and human safety at risk. The increase in hacker attacks during the pandemic on factory networks, with ransomware and other attacks, is a reminder of the security flaws present in many of these environments that need to be fixed and traditional cloud computing is struggling to cater acceptable response times for devices operating at the fringes of a network.

All of this brings important questions: Where to store all this information? How to safely communicate between machines? Which information is safe to share? How to decrease latency? How to manage all this?

In contemporary society, we witness a generation deeply engrossed in the pursuit of productivity. A cursory glance at any bustling city street reveals the perpetual state of action: a prime example is someone simultaneously answering emails from their phone while walking home after work. This generation thrives on multitasking and achievement, which explains why the workforce places a premium on effectiveness, productivity, and efficiency.

For any prudent employer, a fundamental question invariably arises: How can workloads be executed with greater efficiency and heightened automation? As most seasoned professionals in the business world recognize, the answer is straightforward: communication is the linchpin of workplace productivity and efficiency, particularly within industrial settings. Whether considering aspects of safety, transparency, execution, or accessibility, effective communication within a factory or business is the cornerstone that empowers organizations to navigate the ever-changing business landscape, amplify productivity, and sustain their competitive edge in today's digital economy. That is why is important to have all the assets networked and available at all times.

So, what is a network? A network is a collection of interconnected devices, such as computers, servers, and other devices that are connected together to share resources and communicate with each other. It allows devices to share information, communicate with one another and access shared different resources such as printers, storage devices, and applications. They can be wired or wireless, physical or virtual, and used for various purposes such as file sharing, remote access, telecommunication, and many more.

Imagine a global network of servers, each with a unique function. Not a physical identity but instead is a vast network of remote servers around the globe which are hooked together and meant to operate as a single ecosystem, communicating between and exchanging data to allow fast and on demand information about the services that are used by a individual - that is what a cloud is.

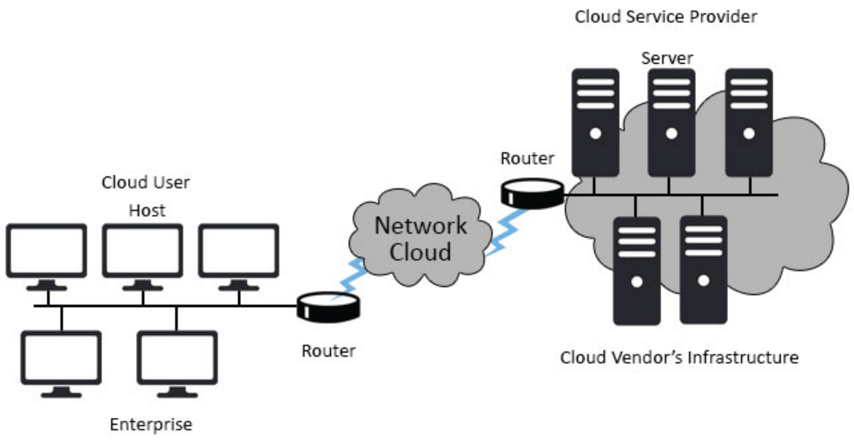


Figure 1.1: Cloud Network

To see the source of each information that arrives to our devices, it's important within a network that each device is unique. The use of IP Address Management (IPAM) - process of planning, assigning, and managing IP addresses within a network - is a critical component of network management, as it ensures that all devices within a network have unique IP addresses and that these addresses are correctly assigned and configured. It can be implemented as a standalone system or integrated with other network management tools such as network monitoring, security, and automation tools. Additionally, IPAM can be integrated with cloud infrastructure for provisioning and management of IP addresses across multiple locations and clouds.

What are the different types of network configurations that are available? How can we ensure seamless communication within an ecosystem? What methods will we employ to store data, and how can we guarantee its immediate accessibility whenever required?

1.1 Objectives

This dissertation presents a framework for computing continuity and edge analytics within a federated system. In the event that a sensor within the factory system identifies a potential component failure, the cloud edge possesses the capability to autonomously deactivate the machinery and issue an alert. Furthermore, this research is dedicated to ensuring scalability by distributing tasks to network-connected sensors and devices, thereby relieving the processing load on central systems while efficiently managing corporate data. Ultimately, all data, whether for internal or external sharing, should be securely stored in the organization's federated data lake.

The following topics outline the tasks and objectives of this dissertation:

- Investigate and analyze various industry approaches to addressing vulnerabilities inherent in the industry's systems. This exploration will provide insights into potential solutions and aid in the integration of a system that effectively mitigates these challenges.

- Define an architectural framework capable of meeting the industry's constraints and requirements.
- Optimize the chosen architectural framework to enhance its efficiency and effectiveness.
- Evaluate the impact of the optimization in real-world scenarios to ensure its practical applicability.

1.2 Motivation

This project is driven by the necessity to enhance response times by minimizing latency in machine decision-making, aimed at reducing the time it takes for machines to make critical decisions. In manufacturing, delays in decision-making can result in inefficiencies, errors, and downtime, by minimizing latency, it makes real-time decisions and responses possible, enhancing the overall agility of the manufacturing process. It also focuses on elevating the efficiency and reliability of these decisions. Efficiency means not only quick decisions but also resource optimization and process streamlining. Reliable decisions are crucial for maintaining product quality and reducing defects. By achieving both efficiency and reliability, the project aims to make manufacturing processes smoother, more cost-effective, and less prone to errors.

In today's world, environmental responsibility is one of the greatest flags to a company. The project endeavors to make a significant contribution to this cause by reducing the ecological impact of businesses. This involves initiatives like decreasing energy consumption, which is not only cost-effective but also environmentally friendly. Lowering bandwidth requirements also contributes to reduced energy consumption and, in turn, minimizes the carbon footprint. Furthermore, the project focuses on streamlining operational and infrastructure expenses, which not only saves costs but also reduces the overall environmental impact. Enhancing the autonomy of machines signifies the ability to make more decisions without human intervention. This can lead to increased operational efficiency and a reduced dependency on manual labor.

As machines become more autonomous and interconnected, the security of the system is of paramount concern. Protecting against cyber threats, ensuring data integrity, and safeguarding sensitive information are critical components. The project incorporates robust security measures, including encryption, access control, and continuous monitoring to ensure that the manufacturing system is resilient against data breaches.

In a manufacturing environment, there may be sensitive data that requires protection, such as proprietary designs, intellectual property, and customer information. Maintaining the privacy of this data is not only essential for compliance but also for building trust with stakeholders. The project aims to implement privacy measures that secure sensitive information while still allowing for efficient data sharing and decision-making within the manufacturing system. [2]

In summary, this project is a comprehensive initiative aimed at enhancing every aspect of machine decision-making within a manufacturing context. Moreover, it seeks to improve the autonomy, security, and privacy of machines operating within a manufacturing system.

CHAPTER 2

State Of The Art

2.1 The industry perspective: Connectivity to increase availability

A network mesh is a type of network topology in which each node or device in the network is connected to multiple other nodes, rather than just one, where each device acts as a relay, forwarding data to other devices in the network. This allows for multiple paths for data to travel, making the network more resilient and fault-tolerant. There are two main categories: full mesh and partial mesh. In a full mesh network, each device is directly connected to every other device in the network. This provides the highest level of fault tolerance, as data can be routed around any failures in the network. In a partial mesh network, not all devices are directly connected to every other device, but there are still multiple paths for data to travel.

Mesh networks are often used in wireless networks, where devices can move around and the network topology can change dynamically. By using a mesh network, devices can automatically find the best path to send data, even if one path becomes unavailable. They are also used in IoT networks, where devices are spread out over a large area and may not be in range of a central hub or gateway. In this case, the mesh network allows devices to communicate with each other and pass data along to the central hub or gateway. [1]

A federated data fabric is a network of interconnected data sources that can be accessed and used by different departments and systems, without compromising data privacy or security. It allows for standardisation, integration, and optimisation of data. A network fabric is "the mesh of connections between network devices such as access points, switches, and routers that transports data to its destination. "Fabric" can mean the physical wirings that make up these connections, but usually it refers to a virtualized, automated lattice of overlay connections on top of the physical topology". [2]

[1]https://www.techtarget.com/iotagenda/definition/mesh-network-topology-mesh-network (accessed 11/05/2023)

[2]https://www.cisco.com/c/en/us/solutions/enterprise-networks/what-is-a-network-fabric.html (accessed 12/05/2023)

2.2 A cloud vs. edge approach

IoT data collection involves the use of sensors to track the performance of devices connected to the IoT. The sensors track the status of the IoT network by collecting and transmitting real-time data that can be stored and retrieved at any time. There are two solutions for storing data: cloud computing and edge computing. It is important to understand that these two concepts are distinct and cannot replace each other: one is used for processing data that is not time-dependent, while the other is used for processing time-critical data.

Cloud computing is a computing technique where IT services are provided by massive low-cost computing units connected by Internet Protocol (IP) networks. Today, the term cloud computing describes the abstraction of web-based systems. Often these cloud-based resources are viewed as virtual, meaning that if a system needs more resources, they can simply be added on demand and usually transparently to the app that uses those resources. Through their virtual nature, cloud-based solutions can be scaled up or down in size, and the companies whose solutions reside in the clouds normally pay only for the resources they consume. Thus, the companies that once relied on expensive data centres to house their processing resources can now shift their costs and maintenance efforts to cheaper, scalable alternatives. There are 5 major technical characteristics of cloud computing: large-scale computing resources, high scalability and elasticity, shared resource pool (virtualized and physical resource), dynamic resource scheduling and general purpose. [3]

Traditional Cloud Computing follows a centralised scheme where computing and storage are deployed in a remote data centre. However, this approach experiences significant limitations when dealing with these emerging technologies that require a real-time response and reduced latency.

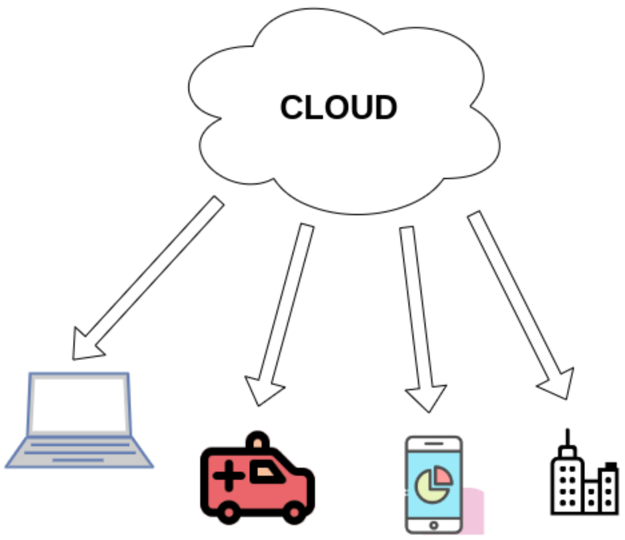


Figure 2.1: Cloud Computing

The IoT services and cloud computing varies across industries, but in recent years, these

[3]https://cloud.google.com/learn/what-is-cloud-computing (accessed 15/05/2023)

technologies have become increasingly important in many sectors, such as:

- Manufacturing: In this sector, the integration of IoT enables the real-time monitoring and optimization of manufacturing processes, including equipment health, quality control, and supply chain management. Cloud computing complements this by providing a robust platform for data storage and analysis.
- Healthcare: The healthcare industry leverages IoT for remote patient monitoring, medical device management, and asset tracking. Cloud services play a pivotal role by securely storing and processing sensitive patient data, ensuring compliance with stringent regulations.
- Agriculture: Within agriculture, IoT technology is harnessed to maximize crop management and livestock monitoring. Cloud computing processes data collected from various sources, including sensors, weather stations, and satellite imagery, enhancing precision agriculture practices.
- Transportation and Logistics: In the transportation and logistics domain, IoT devices are instrumental in tracking vehicles and cargo, monitoring driver behavior, and streamlining fleet management. Cloud computing empowers organizations to manage and analyze data, facilitating route optimization, fuel efficiency, and real-time tracking.
- Smart Cities: IoT sensors and devices serve as the foundation for monitoring and managing urban infrastructure in smart cities. Cloud computing plays a central role by enabling centralized data processing, supporting functions like smart traffic lights, waste management, and environmental monitoring, which benefit city planners and residents alike.
- Energy and Utilities: IoT technology assists in the monitoring and control of energy infrastructure, encompassing smart grids and renewable energy systems. Cloud computing enables the analysis of energy data, grid management, and predictive maintenance.
- Telecommunications: IoT seamlessly integrates with network infrastructure to facilitate device management and monitoring in the telecommunications industry. Cloud computing plays a pivotal role in virtualization and network functions, enhancing scalability and flexibility.

These use cases underscore the versatility and significance of cloud computing across a myriad of industries, driving efficiency, innovation, and competitiveness. With the increase of demand on all this systems several setbacks emerged such as [4]:

- Downtime: Cloud service providers can experience outages or downtime, which may disrupt business operations. Reliability depends on the provider's infrastructure and Service Level Agreement (SLA).
- Security Concerns: Storing data off-site in the cloud can raise security concerns. Data breaches or unauthorized access are potential risks.
- Data Privacy: Data is stored on servers owned and managed by the cloud provider, raising concerns about data privacy and compliance with industry-specific standards regulations.

[4]https://www.investopedia.com/terms/c/cloud-computing.asp (accessed 16/05/2023)

- Bandwidth Limitations: Accessing and transferring large volumes of data to and from the cloud can strain network bandwidth, leading to latency and performance issues.
- Cost Overruns: While cloud computing can be cost-effective, it's essential to monitor usage and costs carefully. If not managed properly, costs can escalate beyond budget.
- Limited Control: Users have limited control over the infrastructure and software, which can be a disadvantage for organizations with specific customization or regulatory requirements. Migrating data and applications from one cloud provider to another can be challenging and costly
- Compliance Challenges: Industries with strict regulatory requirements may find it challenging to ensure compliance with cloud providers' policies and procedures.

In this regard, recent researches propose the use of the Edge Computing paradigm as a means of improving Cloud Computing capabilities. Edge computing allows data from IoT devices to be analysed at the edge of the network before being sent to a data centre or cloud. It brings together the core capabilities of networks, computers, storage and applications to provide intelligent services at the network edge - close to the source of objects or data - to meet the critical needs for agile connectivity, real-time services, data optimisation, application intelligence, security and privacy of the digitisation of industry. This solution is essential to increase speed, lower latency, improve network performance and traffic management and greater reliability when the cloud is down. [5]

It is denominated as "Edge" computing because information processing no longer takes place only in centralised or distributed nodes (core), but is performed also in the other extreme (Edges), moving centralised computing processes away from the centre . All the information produced by IoT devices is processed in the extreme, thus releasing computational load from centralised servers, avoiding network traffic overload, and reducing the response time required by new IoT applications. [6]

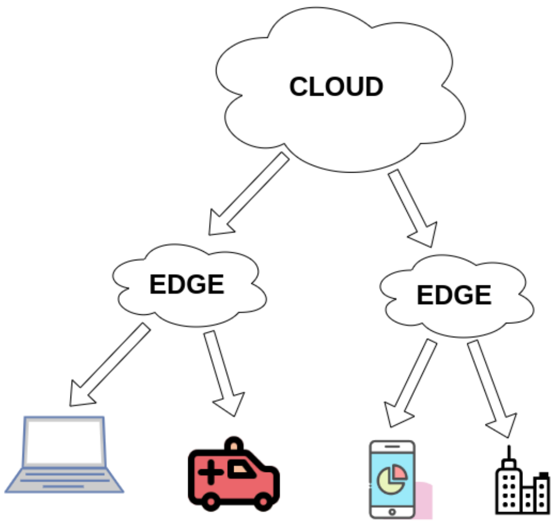


Figure 2.2: Edge Computing

[5]https://ubuntu.com/blog/whats-the-deal-with-edge-computing (accessed 23/05/2023)
[6]https://www.hpe.com/us/en/what-is/edge-to-cloud.html (accessed 23/05/2023

By shifting the processing capacity to the nodes, Edge architectures offer the following advantages:

- Save bandwidth and storage resources, as the rapid increase in IoT devices adoption would choke the bandwidths of existing network infrastructures; to filter noise data streams that come from different data sources are processed by the nodes before sending the data to the Cloud. Proximity and low latency are achieved by processing information close to its source of origin.
- Enhanced scalability is achieved through decentralised storage and processing.The nodes of Edge architectures provide each node of the network with isolation and privacy.

There are different approaches in the edge computing solution:

- the Far-Edge approach where the edge computing infrastructure is deployed in a location farthest from the cloud data centre and closest to the users. It's used for apps that require ultra-low latency, high scalability and high throughput;
- the Near-Edge approach is the edge computing infrastructure which is deployed in a location between the far edge and the cloud data centres. It's used for Content Delivery Network (CDN) caches (globally distributed network of proxy servers deployed in multiple data centers with the goal to serve content to end-users with high availability and high performance) and Fog computing (decentralized computing infrastructure in which data, compute, storage and applications are located somewhere between the data source and the cloud).

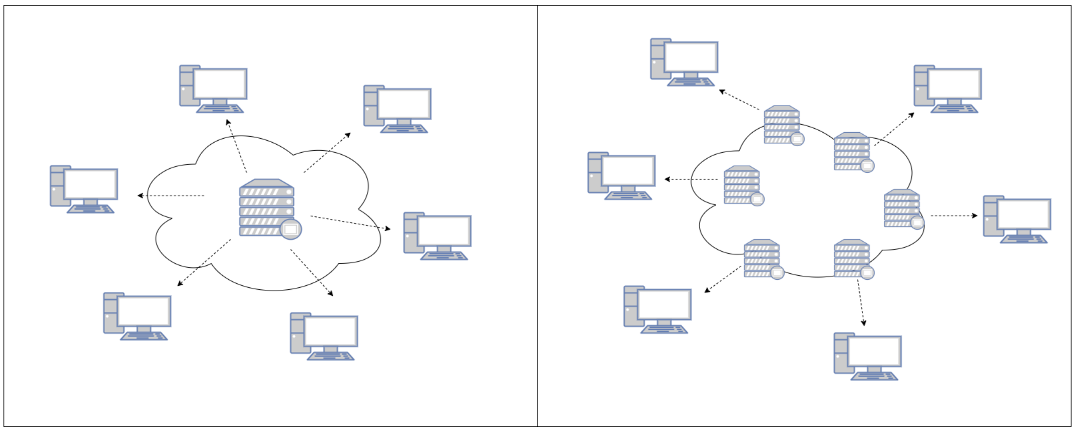


Figure 2.3: Single server distribution vs. CDN distribution

While Far Edge computing infrastructure hosts applications specific to the location in which it is deployed, Near Edge hosts generic services. There are a number of factors which are evaluated to decide where to host a given application. Some of the criteria are scalability (Number of users, number of devices), type of application (Gaming, Video streaming, web content, social media), Latency requirement of the application, throughput requirement for the application, entity that manages the application (Enterprise, Telco, Cloud Service provider) and security constraints.[7]

[7]https://tech.ginkos.in/2019/06/far-edge-vs-near-edge-in-edge-computing.html (accessed 12/05/2023)

Software-Defined Wide Area Network (SD-WAN) is a technology that allows businesses to use software to manage and optimize the network connections between their branch offices and data centers. It allows for real-time monitoring and optimization of network traffic, improving application performance and reducing costs by dynamically directing traffic over the best available path. [8]

By using SD-WAN, an organization can control the flow of data traffic and optimize the use of available bandwidth, which can help to reduce the latency of edge computing applications and improve their performance. Additionally, SD-WAN can improve security and reduce costs by directing traffic over the best available path, which can be particularly beneficial when using edge computing to process sensitive data.

Due to the rapid development of computing technologies, the amount of data collected by edge terminals devices has increased [9]. There are some important factors to consider when it comes to protecting the information, such as the security of the data itself and unauthorised access and use of the data. When it comes to protecting sensitive and private data, data encryption is very important as it can help prevent unauthorised access to the data and ensure that communication between servers and client applications is secure. Enhance the security is one of the practices to secure IoT devices and the network that these devices use. Its primary goals are to preserve user privacy and data confidentiality, ensure the security of devices and other related infrastructure, and allow the smooth functioning of the IoT ecosystem.

Some possible attacks can be done by unauthorized access in entities influencing authentication, authorization and access control, by attacks in transmitted or stored data, influencing data confidentiality and integrity, or even by attacks in communication channels, influencing communication channel protection.

Network mesh and SD-WAN are related in the sense that they all aim to improve the performance, reliability, and security of the network. A network mesh provides redundancy and increased reliability by allowing multiple paths for data to travel, and SD-WAN optimizes the use of available bandwidth and improves the security of the network by directing traffic over the best available path.

2.3 MEC vs FOG

The ETSI has developed a new paradigm called Multi-Access Edge Computing MEC to address the challenges posed by the complexity of rapidly evolving wireless and mobile communications networks. To mitigate the inherent limitations of the current cloud infrastructure, the fundamental idea of MEC is to extend the capabilities of cloud computing to the edge of the mobile network.

MEC is an architecture that brings computing capabilities closer to the edge of the network, allowing data to be processed and analyzed at the point of origin, rather than in a central data center. This architecture is intended to provide low-latency, high-bandwidth services

[8]https://www.cisco.com/c/pt$_{pt}/solutions/enterprise - networks/sd - wan/what - is - sd - wan(accessed$12/05/2023)

[9]https://iot-analytics.com/number-connected-iot-devices/ (accessed 25/04/2023)

to applications that require real-time data processing, such as augmented reality, virtual reality, autonomous vehicles and real-time analytics. MEC also enables the integration of various technologies and services to support various use cases such as IoT, 5G, and Artificial Inteligent (AI). It allows multiple service providers to coexist and provide their services on the same infrastructure, through the use of open Application Programming Interface (API)'s and standard interfaces.

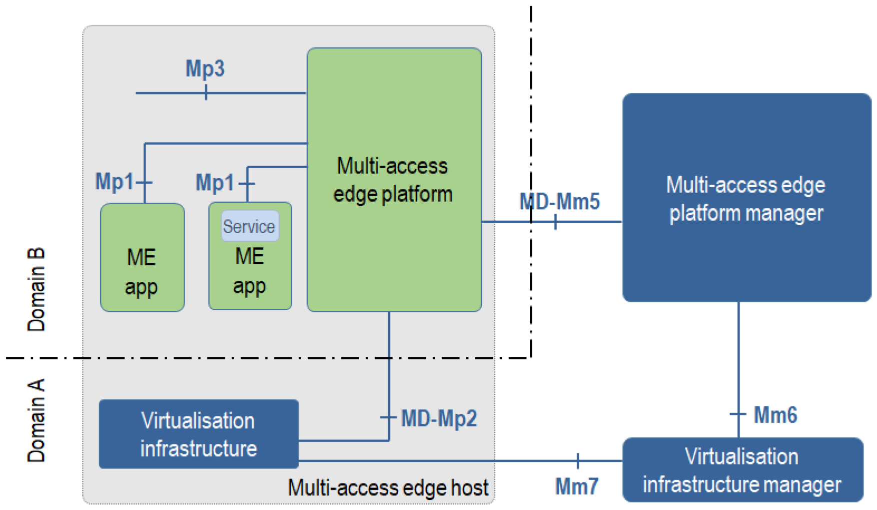

Mp3
Mp1
Mp1
Service
ME app
ME app
Multi-access edge platform
MD-Mm5
Multi-access edge platform manager
Domain B
Domain A
Virtualisation infrastructure
MD-Mp2
Mm6
Virtualisation infrastructure manager
Multi-access edge host
Mm7

Figure 2.4: ETSI MEC reference architecture [3]

The layers of this architecture can be divided into [4]:

- The User Plane: directly interacts with the end-users and devices. It provides low-latency and high-bandwidth services such as content caching, data analytics, and device management.
- The Control Plane: responsible for managing the resources and traffic of the User Plane. It includes functions such as traffic management, mobility management, and service orchestration.
- The Management Plane: responsible for configuring, monitoring, and maintaining the MEC infrastructure. It includes functions such as network management, security management, and performance management.

The MEC architecture is composed of several key components [5]:

- Edge servers: These are small, low-power servers that are deployed at the edge of the network, close to the end-users. They are responsible for running the applications and providing computing resources to the edge devices.
- Edge gateways: These are devices that act as a bridge between the edge servers and the core network. They provide connectivity and routing capabilities, as well as security features such as firewalls and intrusion detection systems.
- Edge applications: These are the applications that are designed to run on the edge servers and take advantage of the low-latency and high-bandwidth capabilities of the MEC architecture.

- Cloud-based services: The MEC architecture is often integrated with cloud-based services, such as storage and analytics services, to provide additional functionality and scalability to the edge applications.
- Network Functions Virtualization (NFV) and Software-Defined Networking (SDN) technologies: These technologies are used to create a flexible and programmable infrastructure that enables the deployment of new services and applications with minimal effort.

In contrast, Fog Computing uses local Fog Nodes to enable local computing. Fog computing is "a system-level horizontal architecture that distributes resources and services of computation, storage, control, and networking anywhere along the continuum from cloud to objects," [10] according to the OpenFog Consortium. Low latency and real-time analytics are advantages that Fog Computing shares with other edge computing types (such as MEC), but storage capacity is limited.

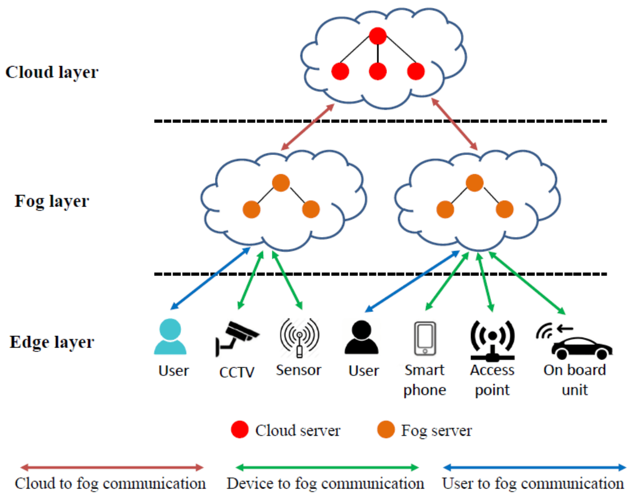


Figure 2.5: FOG Computing architecture [6]

Fog computing is a decentralized computing infrastructure in which data, compute, storage, and applications are distributed in the most logical, efficient place between the data source and the cloud. This can include devices such as routers, switches, and gateways. MEC, on the other hand, is a standardized architecture for edge computing in networks. It involves running applications and services on servers located at the edge of the network, closer to end users. [7]

MEC is an industry-standard defined by the ETSI and is built on top of the existing network infrastructure, being tightly integrated with the mobile network and can access information such as user location, device type, and network status. Fog computing is not an industry standard, and there are multiple vendors and open-source projects that provide fog computing solutions. While fog computing can be integrated with any type of network, it does not have the same level of integration as MEC. [8]

[10]https://www.iiconsortium.org/pdf/OpenFog-Reference-Architecture-Executive-Summary.pdf (accessed 23/05/2023)

[11]

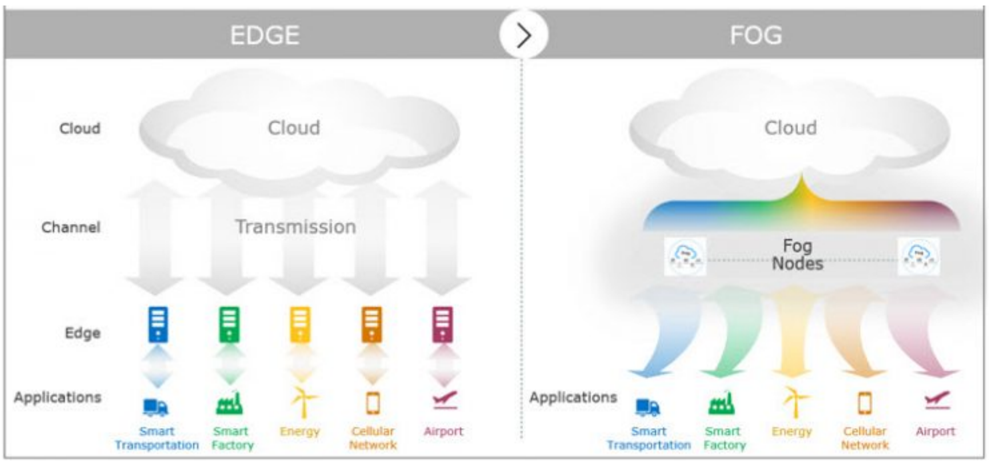


Figure 2.6: FOG vs MEC Architecture

2.4 Types of Messaging protocols

Various types of connections are used in computer networking and the Internet to facilitate communication between devices and systems. Each type of connection is designed for specific use cases and has its own protocols and characteristics. [9] Here are some common types of message protocols:

- **Hypertext Transfer Protocol (HTTP)**: Used for transmitting data over the World Wide Web, it is the foundation of data communication on the internet.
- **Hypertext Transfer Protocol Secure (HTTPS)**: A secure version of HTTP that uses encryption to protect data transmission, commonly used for secure web browsing.
- **Simple Mail Transfer Protocol (SMTP)**: Used for sending email messages, and it's a fundamental protocol for email communication.
- **Post Office Protocol version 3 (POP3)**: Used for retrieving email from a mail server, often employed by email clients.
- **nternet Message Access Protoco (IMAP)**: Another protocol for email retrieval but allows for managing email on the server itself, widely used in modern email clients.
- **Transmission Control Protoco (TCP)**: Part of the TCP/IP suite, it ensures reliable data transmission across networks.
- **User Datagram Protocol (UDP)**: Also part of the TCP/IP suite, it provides a connectionless protocol for faster data transmission, often used for real-time applications.
- **File Transfer Protocol (FTP)**: Used for transferring files between a client and a server on a network.
- **Secure Shell) (SSH)**: Provides secure remote access to a computer over a network, often used for secure system administration.

[11]http://www.codingsoho.com/zh/blog/differences-between-cloud-fog-and-mec-removing-the-mist/ (accessed 18/05/2023)

- **MQTT**: A lightweight and efficient protocol for publish-subscribe communication, commonly used in IoT.
- **Advanced Message Queuing Protoco (AMQP)**: A messaging protocol for applications to communicate with each other, often used in enterprise environments.
- **WebSockets**: A communication protocol that enables bidirectional, real-time communication between clients and servers, commonly used in web applications.
- **Real-time Transport Protocol (RTP)**: Primarily used in delivering audio and video over IP networks, crucial for streaming and VoIP applications.
- **Session Initiation Protoco (SIP)**: Used for initiating, maintaining, modifying, and terminating real-time sessions that involve video, voice, messaging, and other communications applications.
- **Bluetooth**: A wireless communication protocol commonly used for short-range connections between devices, such as smartphones, headphones, and IoT devices.

These are just a few examples of the types of connections used in networking and communication. The choice of connection type depends on the specific requirements of an application, including factors like security, reliability, speed, and the nature of the data being transmitted.

2.5 Why we should save data?

Saving data in edge computing has several advantages, which make it a valuable approach in various applications and scenarios. Edge computing refers to the practice of processing and storing data closer to the source of data generation, typically at or near the "edge" of the network, rather than sending all data to centralized data centers or the cloud.

Saving data in edge computing can improve performance, efficiency, and security in various applications by processing and storing data closer to the source. This approach is particularly valuable in scenarios where low latency, bandwidth efficiency, and real-time decision-making are critical. [10]

So, why we should save data? [11]

- **Data Organization:** Databases provide a structured and organized way to store and manage data, making it easier to maintain and retrieve information.
- **Data Integrity:** Relational databases, in particular, enforce data integrity through constraints, ensuring the accuracy and consistency of data.
- **Data Retrieval:** Databases support efficient data retrieval and querying, allowing users to find and access information quickly.
- **Concurrency Control:** Databases manage concurrent access to data, ensuring that multiple users can work with the data simultaneously without conflicts.
- **Scalability:** Many databases offer scalability options, making it possible to handle large volumes of data and high traffic loads.
- **Data Security:** Databases can enforce access control and provide security features to protect sensitive data.

- **Data Redundancy:** Databases allow for data redundancy and replication for backup and disaster recovery purposes.
- **Atomicity, Consistency, Isolation, and Durability (ACID) Compliance:** Relational databases ensure transactions are Atomic, Consistent, Isolated, and Durable, which is critical for applications that require strong data consistency.
- **Real-time Analytics:** Some databases, like NoSQL and column-family databases, are well-suited for real-time analytics and big data processing.

There are several types of databases, each designed to handle specific data storage and retrieval needs. Databases are essential for managing, organizing, and accessing data efficiently. Here are some common types of databases and why we use them: [12]

[12]https://www.geeksforgeeks.org/types-of-databases/ (accessed 25/05/2023)

Database Type	**Examples**	**Characteristics**	**Use Cases**
Relational Databases	MySQL, PostgreSQL, Oracle, SQL Server	Structured schema with tables, rows, and columns. Well-defined relationships.	Financial systems, transactional applications
NoSQL Databases	MongoDB, Cassandra, Redis, DynamoDB	Designed for unstructured or semi-structured data. High read and write scalability.	Big data, real-time analytics
Document Databases	MongoDB, CouchDB, RavenDB	Store data in JSON or similar format within documents. Flexible schema and hierarchical data structures.	Content management, catalogs
Key-Value Stores	Redis, DynamoDB, Riak	Store data as key-value pairs. Fast read and write operations.	Caching, session management, real-time data processing
Column-family Databases	Apache Cassandra, HBase	For horizontal scalability. High write throughput and analytical workloads.	Applications w/ high write throughput, analytics
Graph Databases	Neo4j, Amazon Neptune	Complex relationships.	Social networks, recommendation systems, network analysis
Time-Series Databases	InfluxDB, Prometheus	Storing and querying time-stamped data.	Sensor data, logs, monitoring metrics
In-Memory Databases	Redis, Memcached	Store data in RAM for extremely fast data access.	Caching, session management, real-time data processing

Table 2.1: Types of databases

2.6 How can we distribute workload across resources?

Load balancing is a crucial technique used in computer networking and server management to distribute network traffic or workload across multiple servers or resources. Its primary purpose is to ensure that no single server or resource is overwhelmed with traffic, thus improving the availability, reliability, and performance of a network or application. Here are some key aspects of load balancing:

- Load balancing is typically achieved using a dedicated device or software known as a load balancer. It sits between the clients (users or applications) and the servers, and it directs incoming requests to the most appropriate server based on a set of predefined algorithms.

- Use various distribution algorithms to decide how to route incoming requests. Some common algorithms include:
  - Round Robin: Requests are distributed in a cyclical manner to each server in the pool.
  - Least Connections: The load balancer routes requests to the server with the fewest active connections.
  - Weighted Round Robin Servers are assigned different weights, and the load balancer sends more requests to servers with higher weights.
  - IP Hash: Requests are distributed based on the source IP address, ensuring that requests from the same client always go to the same server
- Load balancers continually monitor the health of the servers in the pool. If a server becomes unresponsive or unhealthy, the load balancer can stop sending requests to it until it recovers. This ensures that only healthy servers handle incoming requests.
- In some cases, it's necessary to maintain session persistence. This means that subsequent requests from a client are directed to the same server to maintain session state. Load balancers can be configured to support session persistence when needed.
- Allows for easy scalability by adding more servers to the pool to handle increased traffic. It also provides redundancy, as if one server fails, the load balancer can direct traffic to other healthy servers.
- There are various types of load balancers, including hardware load balancers, software load balancers, and cloud-based load balancers. Cloud providers often offer load balancing services as part of their infrastructure offerings.
- Some organizations use global load balancers to distribute traffic across multiple data centers or geographic regions, ensuring high availability and disaster recovery.
- CDNs are a specialized form of load balancing that cache and serve content (such as images and videos) from edge servers distributed around the world. This reduces latency and improves content delivery speed.
- Eessential for high-traffic websites, applications, and services to ensure they remain responsive and available to users. It plays a critical role in maintaining system reliability and optimizing resource utilization

2.7 Architectures

Scalability, performance, versatility, security and cost-cutting capabilities are all characteristics of systems that are primarily determined by their architecture. An effective and clear architecture is crucial because a software architect makes critical decisions about the software that ultimately determine its overall integrity. While they are some architectures available that meet some of the needs of the problem discussed, only a few cover most of them. A reference architecture incorporates industry-accepted best practices, usually providing guidelines for the method of delivery or specific optimal technologies. They help project managers, software developers, enterprise designers, and IT managers to collaborate and communicate effectively in the implementation of a project, giving answers to frequently asked questions and any doubts that may arise. Consequently, it helps teams avoid the errors and delays that would have occurred otherwise.

Recent research has focused on looking into the ability of the Edge Computing Paradigm to lessen the limitations of Cloud-centric architectures. In this respect, Edge Computing architectures are capable of shifting a portion of the computing capacity that is performed from the Cloud to the nodes located at the Edge of the network. [13]

This section provides an overview of reference architectures created for Edge Computing, shedding light on how certain architectures serve as foundation for others.

The ISO/IEC/IEEE 42010:2011 standard is the principal guide to identifying conventions, principles and best practices for consistent Internet of Things architectures or frameworks. The ISO/IEC/IEEE 42010:2011 facilitates the development of evaluation, communication, documentation and systematic or effective resolution in a reference architecture and it's rather used in most of the architectures of edge computing. [14] In 2022, this norm was updated to a new version. [15]

2.7.1 Edge Computing RA 2.0

In 2016, the Edge Computing Consortium (ECC) was created by several organisations, meanwhile the Chinese Academy of Telecommunications Research (CATR) together with the Ministry of Industry and Information Technology (MIIT) formed the Industrial Internet Alliance (IIA) with the aim of boosting the development of industrial Internet in China.

The Edge Computing Reference Architecture (EC-RA) 2.0 was proposed from the joint work between the ECC and the IIA, and it is based on a horizontal layers model with open interfaces. Vertically, this architecture uses the following services: management, data life-cycle and security, with the aim of providing intelligent services throughout the life cycle. The development of this architecture was based on international standards such as ISO/IEC/IEEE 42010:2011 and it presents Edge solutions and frameworks to industries. The layers include:

- Smart Services: based on a model-driven service framework. Intelligent coordination between service development and deployment is achieved through the Development service framework and the Deployment and operation service framework. These frameworks enable coherent software interface development and automatic implementation and operations.
- Service Fabric : defines the tasks, technological processes, path plans, and control parameters of the processing and assembly phases, implementing fast deployment of service policies and fast processing of multiple types of products.

[13]https://iebmedia.com/technology/edge-cloud/edge-computing-set-to-revolutionize-manufacturing/ (accessed 13/05/2023)

[14]https://www.iso.org/standard/50508.html (accessed 20/05/2023)

[15]https://www.iso.org/standard/74393.html (accessed 20/05/2023)

- Connectivity and Computing Fabric : the Operation, Information and Communications Technology infrastructure is responsible for deploying operations and coordinating between the computational resource services and the needs of the organisation.
- Edge Computing Node: in this layer the intelligent Edge Computing Nodes have real time processing and response capacities, are compatible with diverse heterogeneous connections and the security is integrated into the hardware and software.

2.7.2 Industrial Internet Consortium RA

The Industrial Internet Consortium and the ECC have developed their reference architecture using the ISO/IEC/IEEE 42010:2011. The Industrial Internet Consortium Reference Architecture (IIC-RA) and its three principal tiers are: Edge, Platform and Enterprise and are described as follows:

- Edge layer: collects data from the Edge nodes through a proximity network. The main architectural features of this layer include: breadth of distribution, location, scope of governance and nature of the proximity network. Each of the characteristics will change according to the use case.
- Platform layer: responsible for processing and sending control commands from the third layer (enterprise) to the Edge layer. Its function is to group the processes and analyse the data flows from the Edge layer and the upper layers. It manages the active devices in the Edge layer for data consultation and analysis through domain services.
- Enterprise layer: located in the cloud and runs specific applications such as decision support systems, end-user interfaces, and operations management. This layer generates control commands to be sent to the platform and Edge layers and also receives data flows from those layers.

2.7.3 Far-Edge RA

The FAR-Edge RA has been developed as part of the H2020 FAR-Edge project, as a means of overcoming the challenge that industries face in adopting decentralised automation architectures. The FAR-EDGE RA is aligned to IIC-RA concepts and described from two architectural viewpoints: the functional viewpoint and the structural viewpoint. It is a reference architecture for most of the architectures available today.

2.7.4 GECA

This architecture [12] was designed to support the deployment of solutions that required IoT, Edge and Cloud layers, and securing the information through a blockchain from the moment the data was entered from the IoT nodes. This architecture offers a scalable and secure environment for users to obtain their information in real-time, remotely accessing the applications that are deployed in the Business solution layer, as it was based on the IIC-RA solution. Nevertheless, the initial design did not consider the possibility of using software defined networks or network function virtualization to optimize the use of resources in the Edge-IoT networks.

2.7.5 ISEO

The Intel Smart Edge Open is a platform for Edge AI and IoT that was announced by Intel in 2020. [16]

At the core of its architecture is the Edge Compute Layer, which is responsible for running and executing the applications. This layer is built on top of the Intel Architecture and supports a wide range of edge devices, including IoT devices, robotics, and other embedded systems.

The Edge Connectivity Layer is responsible for providing communication and networking capabilities to the edge devices. It supports multiple communication protocols, including WiFi, Zigbee, and cellular, and provides secure and reliable connectivity between the edge devices and the cloud.

The Edge Management Layer is responsible for device management and provisioning, as well as monitoring and diagnostics. It provides a set of tools and services that can be used to manage, provision, and monitor the edge devices, including a device management service, a cloud-based analytics service, and a software development kit.

Finally, the Edge Security Layer provides security features to the edge devices, including secure boot, secure firmware updates, and data encryption. It ensures the authenticity and integrity of the edge devices and the data they generate, protecting the devices and the data from unauthorized access and tampering.

The fact that this architecture is not open source, brings a lot of disadvantages when researching. Most of the key features are not presented on the developer kit and only on the paid version.

2.7.6 Edge X Foundry

The EdgeX Foundry is an open-source project that provides a standardized and vendor-neutral platform for building and deploying IoT edge computing solutions. Its architecture is designed to be highly flexible and modular, allowing for interoperability and scalability in diverse IoT environments. This architecture is based on the IIC-RA architecture and has MEC compatibility. The key components are:

- Device Services: These components interface with IoT devices and sensors, facilitating data collection and integration.
- Core Services: provide essential functionality such as data normalization, device management, and security.
- Supporting Services: offer extended capabilities, including data export, rules processing, and security functions.
- Application Services: enable the development and deployment of custom applications and analytics at the edge.

[16] https://www.intel.com/content/www/us/en/developer/tools/smart-edge-open/overview.html (accessed 12/05/2023)

2.8 Other architectures worth mention

Several other architectural solutions should be noted in the context of the summary provided in 2.2. However, they are not deemed suitable for our solution for various reasons:

- **Akraino**[17]: Akraino, though noteworthy, is still in the developmental phase, with limited work in the area of edge computing. This ongoing development phase makes it less suitable for our current needs.
- **Thingsboard**[18]: While Thingsboard is an option, it does not adhere to industry standards, potentially leading to interoperability and integration issues with existing systems. Therefore, it may not align with our solution's requirements.
- **Intel-SAP** [19] **and Huawei Edge Gallery** [20]: Both Intel-SAP and Huawei Edge Gallery lack support for robust data encryption. Given the sensitivity of data in edge computing, the absence of data encryption features makes them less suitable for ensuring the security and privacy of data in our solution.

In summary, while these architectural options have their merits, they present specific limitations and challenges that make them less fitting for our particular use case.

2.8.1 Summary of most used architectures

SOLUTION	**EC-RA 2.0**	**IIC-RA**	**GECA**	**Akraino**	**Thinsboard**
Open-Source	No	No	Yes	Yes	Yes
Standards	Yes	Yes	Yes	Yes	no
Data Encription	No	No	Yes	Yes	Yes
Containers/Kubernets Suport (VM support)	Yes	Yes	Yes	Yes	Yes
Scalability	Yes	Yes	Yes	Yes	Yes
Multi Access Edge Computing Compatibility	Yes	Yes	Yes	Yes	No
Connectivity and communication	Yes	Yes	Yes	Yes	Yes
Hardware Acceleration	Yes	Yes	Yes	Yes	Yes
Telemetry and Monitoring	Yes	Yes	Yes	Yes	Yes
Enhanced Security	Yes	Yes	Yes	Yes	Yes
Device Management	Yes	Yes	Yes	Yes	Yes

Table 2.2: Most used architetures (part 1)

SOLUTION	**Edge X Foundry**	**ISEO**	**Huawei Edge Gallery**	**Far-Edge RA**	**Intel-SAP**
Open-Source	Yes	No	Yes	Yes	No
Standards	based on IIC-RA	No	based on IIC-RA	No	No
Data Encription	Yes	No	No	No	No
Containers/Kubernets Suport (VM support)	Yes	Yes	Yes	Yes	No
Scalability	Yes	Yes	Yes	Yes	Yes
Multi Access Edge Computing Compatibility	Yes	No	yes	No	Yes
Connectivity and communication	Yes	Yes	Yes	Yes	Yes
Hardware Acceleration	Yes	No	Yes	No	No
Telemetry and Monitoring	Yes	Yes	Yes	Yes	Yes
Enhanced Security	Yes	Yes	Yes	Yes	No
Device Management	Yes	Yes	Yes	Yes	Yes

Table 2.3: Most used architetures (part 2)

[17] https://wiki.akraino.org/display/AK/Akraino+Edge+Stack+Goal+and+Key+Principles (accessed 12/05/2023)

[18] https://thingsboard.io/docs/ (accessed 12/05/2023)

[19] https://www.sap.com/products/erp/rise.html (accessed 12/05/2023)

[20] https://www.edgegallery.org/en/pc-edgegallery-overview/ (accessed 12/05/2023)

CHAPTER 3

Architecture and Implementation

3.1 Use Case

A manufacturing company has several factories located in different parts of the world. Each factory has a large number of machines and equipment that generate vast amounts of data, such as sensor readings, production statistics, and maintenance information. The company wants to leverage this data to optimize its operations and improve efficiency, but it faces several challenges. The data is generated in real-time and is highly distributed, which makes it difficult to collect, process, and analyze it centrally. Moreover, some of the factories may have limited connectivity, which can cause delays and disruptions in data transfer.

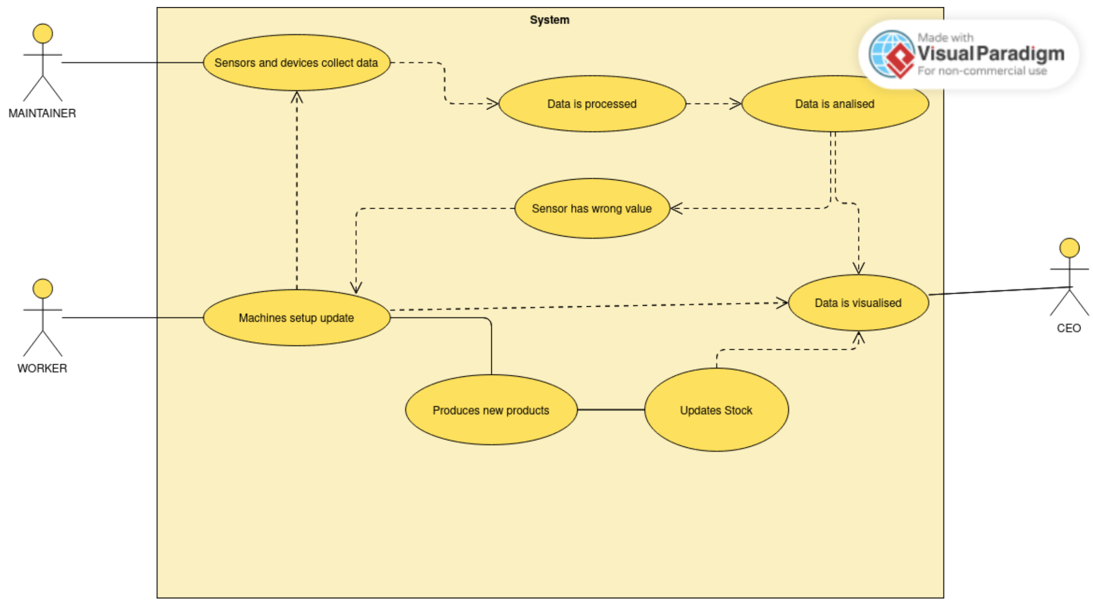


Figure 3.1: Use case

Sensors and devices at the edge of the network collect data from various sources, such as machines, equipment, and environmental sensors. The data is processed at the edge of the network and applications that leverage the data collected are processed by Edge X

Foundry. The data is visualized and analyzed using tools such as dashboards, reports, and analytics platforms. Some factories may require more advanced analytics or machine learning capabilities, which could be implemented using additional software components or integrations with third-party platforms. Other factories may require more specialized data collection or processing tools, such as those for high-speed data streams or industrial protocols. In such cases, the basic flow would need to be customized to accommodate these requirements. [13]

3.2 Requirements

Based on the previous section, we can extract important requirements of the test case. These requirements should serve as a foundation for the development and implementation of the system that will help the manufacturing company leverage its data for optimizing operations and improving efficiency.

3.2.1 Functional Requirements:

- The system should collect data from various sources, including machines, equipment, and environmental sensors.
- Data collected from the sources should be processed in real time and available in the most quickly possible way.
- The system should provide tools for visualizing the processed data, including dashboards, reports, and analytic platforms.
- The system should support three user roles: workers, maintainers, and factory CEO's. Each role has specific access and functionality requirements.
- Workers should be able to check the status of each machine and monitor the product manufacturing process, whenever needed.
- Maintainers should be able to check the vital status of each machine and make decisions based on this information, whenever needed.
- Factory CEO's should be able to access information about the current state of each building, including machine status, production statistics, and more, whenever needed.
- The system should detect and respond to sensor failures as a trigger event.
- The system should detect and respond to missing stock as a trigger event.
- The system should detect and respond to the absence of workers in the building as a trigger event in case some sensor triggers some emergency.

3.2.2 Non-Functional Requirements:

- The system should be capable of handling highly distributed data generated by machines located in different parts of the world.
- The system should be able to handle limited connectivity situations at some factories to prevent delays and disruptions in data transfer.
- The system should be customizable to accommodate the specific requirements of different factories. This includes the ability to implement advanced analytics, machine learning capabilities, and support for specialized data collection or processing tools.

- The system should ensure the security and integrity of the collected and processed data.
- The system should be scalable to handle a large number of machines and equipment generating vast amounts of data.
- The system should perform efficiently to optimize manufacturing operations and improve overall efficiency.
- The system should provide monitoring and alerting capabilities to notify relevant stakeholders about trigger events and abnormal conditions.
- The system should be able to integrate with third-party platforms when advanced analytics or machine learning capabilities are required.
- The system should be user-friendly and provide an intuitive interface for the three user roles (workers, maintainers, and factory CEO's).
- The system should be reliable to ensure uninterrupted data processing and analysis.

3.3 To MEC or not to MEC

MECand fog computing are both edge computing paradigms designed to bring computation and data processing closer to the source of data and reduce latency. However, there are several reasons to choose MEC over fog computing regarding this requirements:

- MEC is typically deployed closer to the mobile network infrastructure, making it an ideal choice for applications that require low latency and high-speed data processing. Fog computing, on the other hand, may have a more dispersed and variable network topology.
- MEC is tightly integrated with 4G/5G networks, enabling seamless and efficient communication with mobile devices. This integration can provide superior support for mobile applications, such as IoT devices.
- MEC benefits from standardized interfaces and API developed by organizations like the ETSI. This standardization can simplify the development and deployment of MEC applications, ensuring interoperability across different vendors.
- MEC can offer stringent Quality of Service (QoS) guarantees, which are crucial for applications like autonomous vehicles, industrial automation, and critical healthcare systems. The proximity to the mobile network infrastructure allows for better control over QoS parameters.
- MEC can take advantage of network slicing, that allows the creation of virtual network segments tailored to specific applications. This ensures that resources are allocated efficiently for different use cases, optimizing performance.
- MEC can be more scalable when it comes to handling a large number of devices or users in a densely populated area. This scalability is especially important for urban environments and crowded events.
- MEC can be more centrally managed and orchestrated due to its close relationship with the core network infrastructure. This centralized management can simplify resource allocation, application deployment, and updates.

- MEC can benefit from the security features, such as network segmentation and encryption, which can enhance the security of edge applications. The integration with the core network provides more control over security measures.
- MEC allows service providers to have greater control over their edge infrastructure, making it easier to customize and optimize the edge computing environment according to their specific needs and use cases.
- MEC can provide quicker and more responsive services, making it well-suited for applications that require rapid data processing and decision-making, such as real-time analytics.

3.4 Edge X Foundry

A set of guiding principles shaped EdgeX Foundry's architectural design and functionalities from its inception. These guiding concepts are essential to ensuring that EdgeX Foundry can provide a strong, flexible, and safe foundation for a variety of applications while also meeting the changing needs of edge computing. We will examine the fundamental principles of architecture that form the basis of EdgeX Foundry and select it as the architecture for this dissertation [1]:

- **Platform-Agnostic Design**: The primary principle of EdgeX Foundry is platform-agnosticism. It is made to function flawlessly with a wide range of hardware, including ARM and x86 architectures. It also is irrelevant which operating system, distribution, deployment/orchestration techniques, or protocols are used. EdgeX Foundry's widespread usage is ensured by its adaptability, which makes it compatible with a wide range of hardware and software settings.
- **Flexibility and Extensibility**: EdgeX Foundry's extensibility and flexibility are key components of its design. It makes it possible to update, swap out, or improve any platform component with other microservices or software elements. The capacity to adapt is crucial for meeting the dynamic needs of edge computing applications. In addition, EdgeX Foundry facilitates dynamic service scaling according to device capabilities and use case specifics, guaranteeing optimal resource use.
- **Reference Implementations and Best-of-Breed Solutions**: In addition to offering flexibility, EdgeX Foundry provides developers with "reference implementation" services, which act as a jumping off point. The purpose of these reference implementations is to facilitate interoperability and direct the development process. On the other hand, EdgeX Foundry promotes the use of best-of-breed solutions, giving customers the freedom to select the parts that are most suited to their unique requirements.
- **Store and Forward Capability** : EdgeX Foundry has a "store and forward" feature to help with the problems caused by remote and disconnected edge systems in addition to sporadic connectivity. This feature makes it possible to gather, store, and transfer data while connectivity fails and then regained. This guarantees data flow continuity and integrity, which is essential in edge environments with erratic network connections.

[1] https://docs.edgexfoundry.org/2.0/edgex-foundry-architectural-tenets (accessed 15/08/2023)

- **Facilitating Edge Intelligence**: EdgeX Foundry is designed to enable "intelligence" to move closer to the edge, addressing issues related to actuation latency, bandwidth constraints, storage limitations, and remote operations. By empowering intelligent decision-making at the edge, EdgeX Foundry optimizes the efficiency and responsiveness of edge computing applications.
- **Brown and Green Field Deployments**: EdgeX Foundry acknowledges the diverse deployment scenarios in the field, supporting both brownfield (existing) and greenfield (new) deployments of devices and sensors. This accommodates legacy systems and facilitates the integration of cutting-edge technologies into a wide range of environments.
- **Security and Manageability**: Finally, EdgeX Foundry places a strong emphasis on security and ease of management. The framework incorporates robust security measures at every level to protect data and devices, while also offering user-friendly management tools for streamlined configuration, monitoring, and maintenance of edge computing infrastructure.

Areas with poor connection present issues that Edge X Foundry helps to overcome. By processing data at the network edge, it lowers the need for a centralized site to have continuous high-bandwidth access, ensuring data analysis even in the event of network failures. It avoids data loss during connectivity problems by allowing devices to cache data locally, maintaining data retention until network access is restored. It has the ability to rank important data first, guaranteeing that even in places with poor connectivity, vital information is transmitted. With its support for offline operation, data processing and monitoring are made possible even in the absence of a network connection, and data is synchronized upon reconnecting.

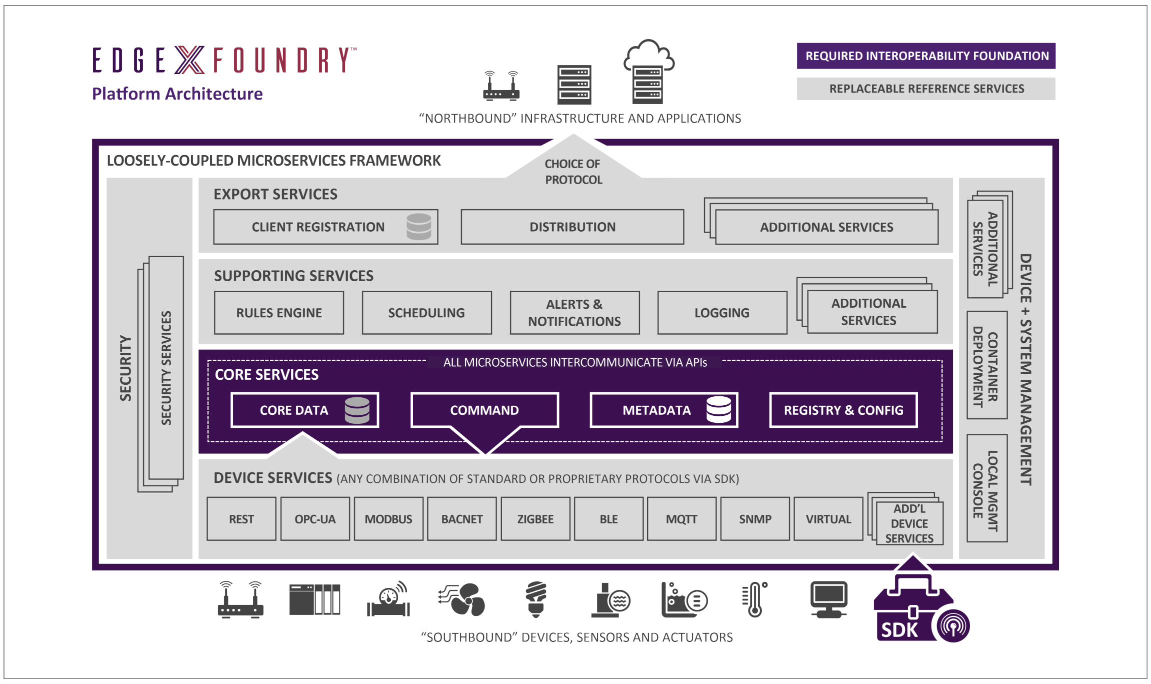


Figure 3.2: Architecture of Edge X Foundry

[2]

EdgeX Foundry is also lot of times called "a collection of open source micro services" [3]. These micro services are organized into 4 service layers and 2 system services.

3.4.1 Service Layers

Core Services Layer

They provide the intermediary between the north and south sides of EdgeX. They are the core to EdgeX functionality, where most of the innate knowledge of what devices are connected, what data is flowing through, and how EdgeX is configured. It consists of:

- **Core data**: a persistent repository for information gathered from devices, along with a management service linked to it.
- **Command**: a service that facilitates and controls actuation requests from the north side to the south side.
- **Metadata**: a repository and associated management service of metadata about the objects that are connected to EdgeX and provides the capability to provision new devices and pair them with their owning device services.
- **Registry and Configuration**: provides other micro services with information about associated services within EdgeX and micro services configuration properties.

Supporting Services Layer

They incorporate a wide range of micro services to include edge analytics - known as local analytics - like ormal software application duties such as scheduler, and data clean up. These services often require some amount of core services in order to function. In all cases, supporting service can be can be left out of an EdgeX deployment depending on use case needs and system resources. They include:

- **Rules Engine**: the reference implementation edge analytics service that performs if-then conditional actuation at the edge based on sensor data collected by the EdgeX instance. This service may be replaced or augmented by use case specific analytics capability.
- **Scheduler**: internal EdgeX “clock” that can kick off operations in any service. At a configuration specified time, the service will call on any EdgeX service to trigger an operation. For example, the scheduler service periodically calls on core data APIs to clean up old sensed events that have been successfully exported out of EdgeX.
- **Alerts and Notifications**: central facility to send out an alert or notification. These are notices sent to another system or to a person monitoring the instance.

Application Services Layer

Application services provide the tools to take sensed data from EdgeX, process it, and send it to a desired endpoint or process. Nowadays, EdgeX includes examples of application

[2]https://docs.edgexfoundry.org/2.0/ (accessed 23/05/2023)
[3]https://docs.edgexfoundry.org/2.0/ (accessed 1-06-2023)

services to send data to HTTP(s) REST endpoints, MQTT(s) topics, and many of the major cloud providers. They rely on the premise of a "functions pipeline" - a group of functions that handle messages in an established sequence.The functions pipeline execution is started by a trigger and the message is then acted upon by each function. Filtering, transformation, compression, and encryption are examples of common functions. Once the message has passed through every function and has been assigned, the function pipeline comes to an end.

Device Services Layer

Device services are the edge connectors interacting with the sensors and devices. They may service one or a number of things or devices at one time and communicates through protocols native to each device object. It converts the data produced and communicated by the IoT object into a common EdgeX data structure, and sends that converted data into the core services layer, and to other micro services in other layers of EdgeX Foundry.

3.4.2 System Services

Security

Security elements of EdgeX Foundry protect the data and control of IoT objects managed by it. There are two major EdgeX security components:

- A security store, which is used to provide a safe place to keep the EdgeX secrets. Examples of EdgeX secrets are the database access passwords used by the other services and tokens to connect to cloud systems.
- An API gateway serves as the reverse proxy to restrict access to EdgeX REST resources and perform access control related works.

System Management

System management provides the primary point of contact for external management systems in order to start, stop, and restart EdgeX services, as well as obtain metrics about the EdgeX services.

3.4.3 How Edge X works and collects data?

EdgeX's primary job is to collect data from sensors and devices and make that data available to north side applications and systems. Data is collected from a sensor by a device service that speaks the protocol of that device. The device service translates the sensor data into an EdgeX event object. The device service can then either:

- put the event object on a message bus (which may be implemented via Redis Streams or MQTT).
- Send the event object to the core data service via REST communications.

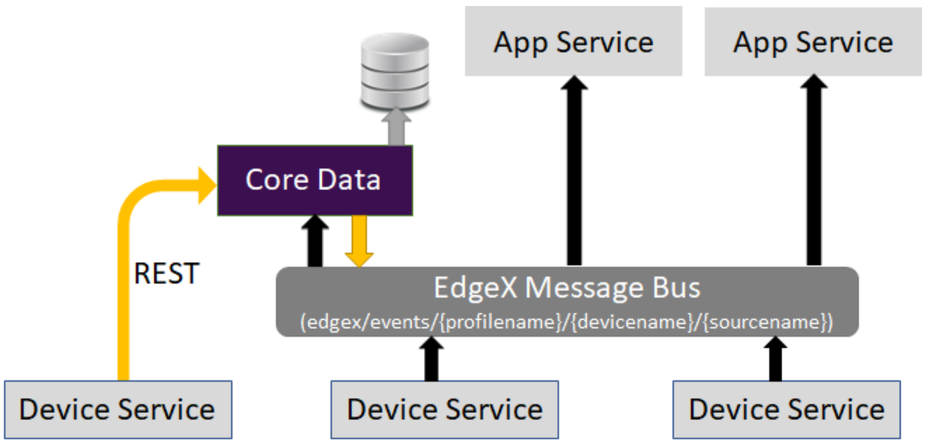


Figure 3.3: Edge X Foundry data flow

[4] When core data receives the event it persists the sensor data in the local edge database. EdgeX uses Redis as our persistence store. Data is persisted in EdgeX at the edge for two basics reasons:

- Edge nodes are not always connected. During periods of disconnected operations, the sensor data must be saved so that it can be transmitted northbound when connectivity is restored. This is referred to as store and forward capability.
- In some cases, analytics of sensor data needs to look back in history in order to understand the trend and to make the right decision based on that history.

When core data receives event objects from the device service via REST / MQTT, it will put sensor data events on a message bus destined for application services, then send the event object to the core data service via REST communications. Finally, the core data will put sensor data events on a message topic destined for application services.

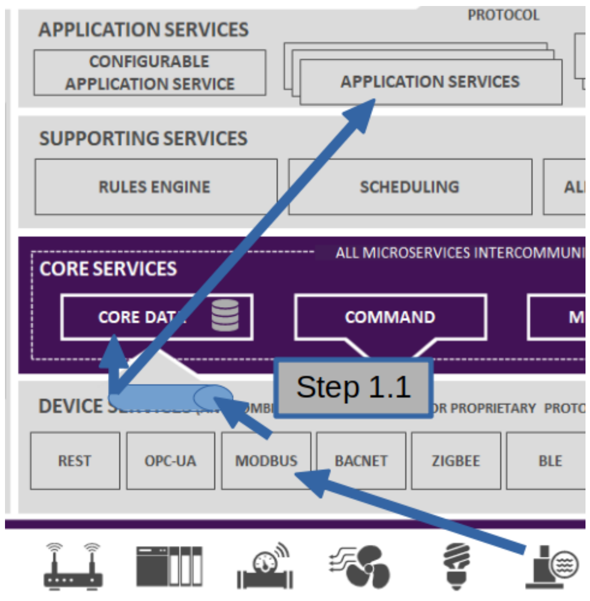


Figure 3.4: Step 1

[4]https://docs.edgexfoundry.org/2.0/ (accessed 25/07/2023)

5

Merely gathering sensor data is only one aspect of an edge platform's responsibility in edge computing. The ability of an edge platform to locally analyze incoming sensor data and take prompt action based on that analysis is another crucial function.

Local analytics hold significance because certain decisions cannot afford to wait for responses from the cloud because the connectivity can be limited and not always connected, so it allows systems to operate independently, at least for some time, and to respond swiftly and in a low-latent manner when system operations are at stake. In other words, events can be used to trigger action back down on a sensor/device with a rule engine - a software system that examines various elements of the data and monitors the data, and then triggers some action based on the results of the monitoring of the data.

The analytical package has the ability to examine sensor event data and determine when to activate a device. For instance, it could verify that an engine's temperature reading is higher than 80ºC and instructs the core command service to "turn on the air conditioner", when it is found that the rule in question is valid.

After receiving the actuation request, the core command service selects the device on which to perform the actuation and calls the owning device service to carry out the action. Developers can implement extra security measures or checks prior to activation with the help of core command.

After receiving the request for actuation, the device service converts it into a request specific to the protocol and sends it the request to the desired device.

[5] https://ackcio.atlassian.net/wiki/spaces/SNA2/pages/128253954/How+EdgeX+Works (accessed 25/07/2023)

3.5 Architecture of the solution

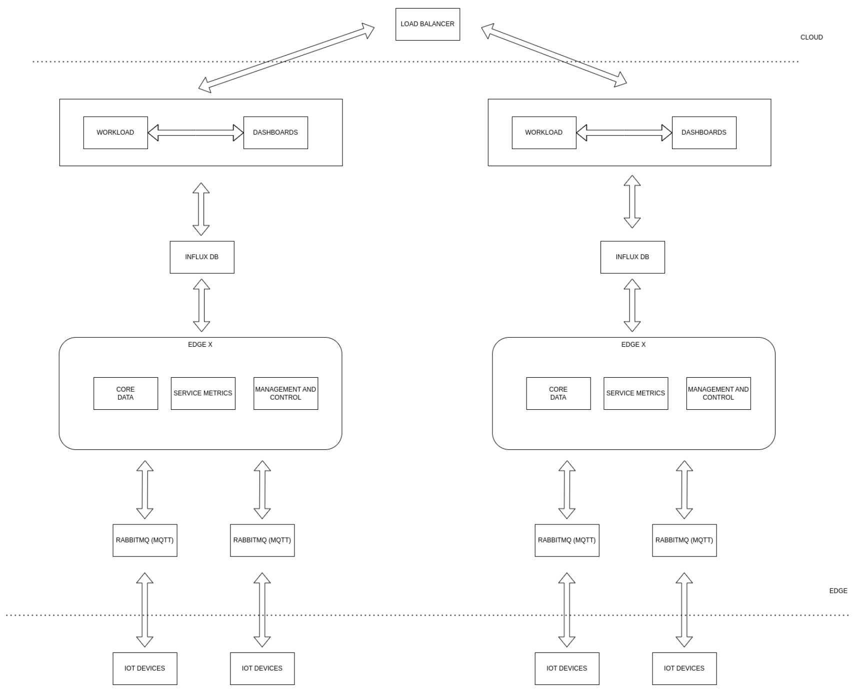


Figure 3.5: Architecture

3.5.1 Security

EdgeX Foundry micro services use a variety of secrets and make use of the secret store to generate, save, and retrieve secrets related to those micro services. [6]

Currently the EdgeX Foundry secret store is implemented with Vault (open source software product)- a tool that securely accessing secrets, such as API keys, passwords or database credentials. The Security Services must create a Secret Store for each Device Service instance that is operating in secure mode, and must add the redisdb known secret to their SecretStore in order for any Device Service to connect to the message bus.

3.5.2 Databases

The decision to use Redis inside the Edge X architecture and InfluxDB outside of it can be attributed to the specific strengths and use cases of each database technology, and the architectural requirements of the Edge X framework. Here's why this configuration was picked:

- **Redis Inside Edge X**:[7]
  - An in-memory data store, which means it excels at storing and retrieving data quickly. In an edge computing environment, where low latency and high-speed data processing are critical, Redis is an excellent choice for caching frequently accessed data and for real-time data processing. It can store data in memory, ensuring rapid access without disk I/O delays.

[6]https://docs.edgexfoundry.org/2.1/security/Ch-SecretStore/ (accessed 24/07/2023)
[7]https://redis.io/docs/management/sentinel/ (accessed 20/06/2023))

  - Includes publish/subscribe messaging capabilities, making it well-suited for real-time communication and event-driven architectures. This is beneficial in Edge X for quickly disseminating data and events among edge devices and modules.
  - Particularly efficient for tasks like data filtering, transformation, and quick decision-making. It can help manage and process data at the edge without the need for frequent disk writes or reads.
- **InfluxDB Outside Edge X**:[8]
  - Designed specifically for handling time-series data, which is common in IoT and edge computing scenarios. It is optimized for storing and querying data points that have a timestamp associated with them. Keeping InfluxDB outside of Edge X ensures that historical data is efficiently stored and can be accessed for analysis and reporting.
  - Scalable and can handle large volumes of time-series data. By placing it outside the edge environment, you can ensure that historical data can be retained and analyzed over extended periods. Storing this data within the edge environment might be resource-intensive and less cost-effective.
  - Can serve as a central repository for data from multiple edge nodes, making it easier to manage and analyze data from across the edge network. This centralized approach simplifies data storage and retrieval for analytics and reporting applications.

By utilizing Redis inside the Edge X framework and InfluxDB outside it, you strike a balance between real-time, in-memory data processing and long-term, efficient storage and analysis of time-series data. The connection between both is Telegraf - an open-source agent for collecting and reporting metrics and data, and it can be configured to subscribe to messages from message brokers like MQTT and then forward that data to InfluxDB for storage and analysis.

3.5.3 MQTT

Optimizing communication between devices and EdgeX was essential to ensure a strategy could be implemented in case of system failures and to safeguard valuable data. The next step involved integrating a message broker into the system to enable a large number of devices to communicate effectively. Instead of using real devices, we employed a script to simulate a custom-defined MQTT device. This approach offers a convenient means to test device-MQTT functionality through an MQTT broker and facilitates communication with other devices, as illustrated in 3.3. This enhancement allows for values to be seamlessly transmitted to the application service when needed, enabling faster decision-making and reducing latency.

[8]https://docs.influxdata.com/influxdb/v1/ (accessed 20/06/2023))

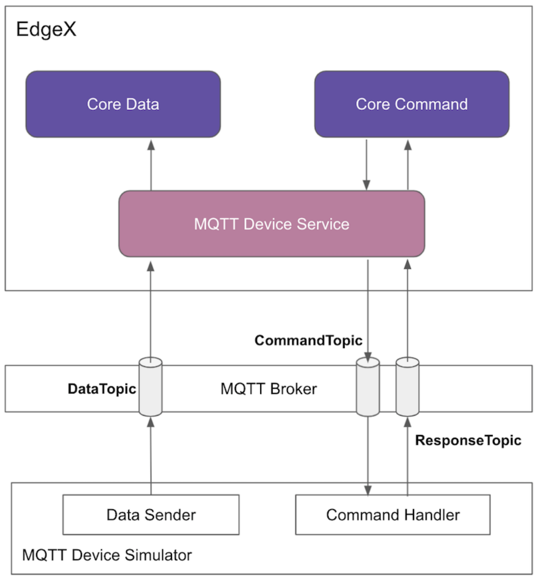


Figure 3.6: MQTT device simulator architecture

[9]

3.5.4 Dashboards

Utilizing Edge X in conjunction with Grafana, Portainer, and a Bootstrap dashboard on the edge offers a comprehensive solution for monitoring and managing devices in an edge computing environment.

Portainer.io : Container Management

Portainer simplifies the management of containers running on edge devices. This is particularly valuable in an edge computing setup, where containers may be distributed across multiple nodes. Its user-friendly interface makes it easier to deploy, monitor, and manage containers without the need for complex command-line interfaces and can offer a centralized view of containerized applications running across different edge nodes. This centralized control simplifies updates, scaling, and troubleshooting tasks. [10]

Grafana: Real-Time Monitoring

Grafana is an excellent tool for creating interactive and real-time data visualizations. It allows you to create dashboards with customizable charts, graphs, and other visual elements to display real-time data from Edge X devices.It provides alerting capabilities, enabling you to set up notifications based on predefined conditions or thresholds. This is crucial for immediate response to anomalies or critical events in the edge environment. [11]

Bootstrap dashboard

A Bootstrap dashboard provides a user-friendly web interface for accessing information about connected devices, as it offers a modern and visually appealing user interface for device

[9]https://docs.edgexfoundry.org/2.0/(accessed 23/07/2023)
[10]https://docs.portainer.io/start/architecture (accessed 23/07/2023)
[11]https://grafana.com/docs/grafana/latest/fundamentals/ (accessed 23/07/2023)

management. The dashboard can display vital information about connected devices, including their status, configurations, and any relevant alerts or notifications. This helps operators and administrators quickly assess the health of the edge environment.

Together, these components empower administrators and operators to efficiently monitor, manage, and respond to the dynamic conditions and events that occur in an edge computing environment, ensuring the reliability and performance of the connected devices.

3.5.5 Load Balancer

Cloud load balancing takes a software-based approach to distributing network traffic across resources, as opposed to hardware-based load balancing, which is more common in enterprise data centers. A cloud load balancer plays a vital role in optimizing the distribution of network traffic between two systems, particularly when Edge X is involved.

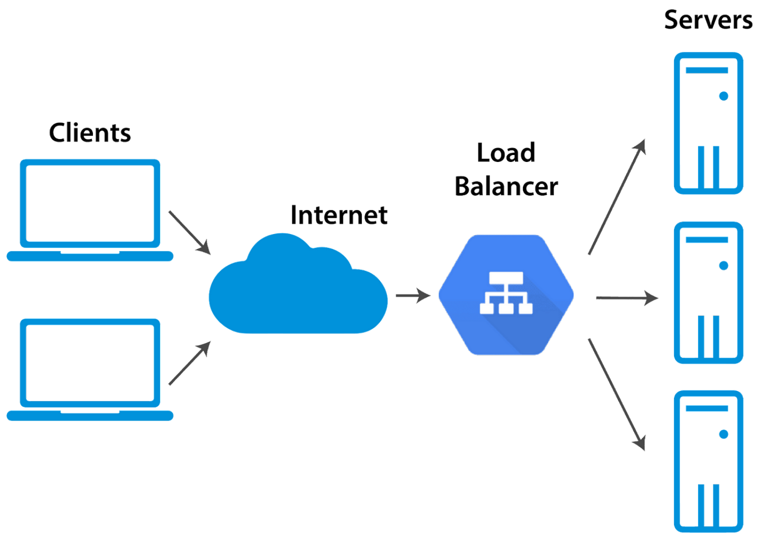


Figure 3.7: Cloud load balancer

12

The primary responsibility of a cloud load balancer is to evenly distribute incoming network traffic between multiple systems or servers. In the context of Edge X, this means balancing the requests and data flows between different edge devices or nodes, ensuring that no single device is overwhelmed with excessive traffic. They monitor the health and performance of the connected systems and assess factors such as the system's current workload, response times, and resource utilization. These metrics helps it make intelligent decisions about where to route incoming traffic, to provide redundancy and failover mechanisms: In the case of Edge X, if one edge device or system becomes unavailable due to a failure or maintenance, the load balancer can automatically redirect traffic to healthy devices. This ensures uninterrupted service and high availability and scale the distribution of traffic based on the current demand.

As traffic increases or decreases, the load balancer can direct traffic to additional devices or reduce the load on underutilized ones. They can act as a barrier between the external

[12]https://levelup.gitconnected.com/load-balancing-on-google-cloud-platform-gcp-why-and-how-a8841d9b70c(accessed in 24/07/2023)

network and the internal edge devices, filtering out potentially harmful traffic. In some cases, certain sessions need to be maintained on the same edge device to ensure data consistency, so the load balancer can implement session persistence, ensuring that requests from a specific client are consistently routed to the same device for the duration of the session.

Another important aspect is the monitoring and analytics tools that help administrators gain insights into traffic patterns, system performance, and potential issues. This data is valuable for optimizing the network and diagnosing problems.

CHAPTER 4

Results

The results were obtained using a Raspberry Pi 4 Model B, with the following specifications:

- Central Processing Unit (CPU): Quad-core ARM Cortex-A72 processor, running at 1.5 GHz.
- RAM: 4GB.
- ROM: 32 GB.
- Storage: MicroSD card slot for system storage.
- Operating System Support: Raspberry Pi OS.

4.1 Unlocking the Edge X: The First deploy

Prior to integrating all the interfaces, the performance of the Edge X system underwent testing. The initial test involved evaluating how the system behaved with a single edge communicating with IoT devices. This communication proceeded smoothly, and the system demonstrated its ability to handle a substantial volume of IoT devices exchanging data continuously for some minutes. Initially, REST communications were employed for making HTTP requests. The containers consistently remained operational, and even when they went down, they proved capable of self-regeneration. Portainer played a crucial role in identifying the initial issues in this project by providing insight into which containers were encountering deployment problems.

Subsequently, a series of 24-hour tests were conducted, gradually increasing the number of messages sent by IoT devices each hour. As anticipated, the volume of generated data grew in proportion to the number of messages exchanged with the Edge X. This led to a gradual but not significant decrease in available memory. The system performed well during the initial hours, but issues began to surface concerning the initial setup, which involved a basic configuration of Edge X with no communication rules and constant data consumption. Consequently, some containers started to experience issues.

One significant challenge in device communication was minimizing delays while enabling Edge X to display and consume information effectively. The initial setup suffered from delays as data volume increased because it lacked a queue for displaying information on the edge.

Following the optimization of the message bus and the utilization of MQTT to transmit device information to the edge, there was a notable improvement in latency times.

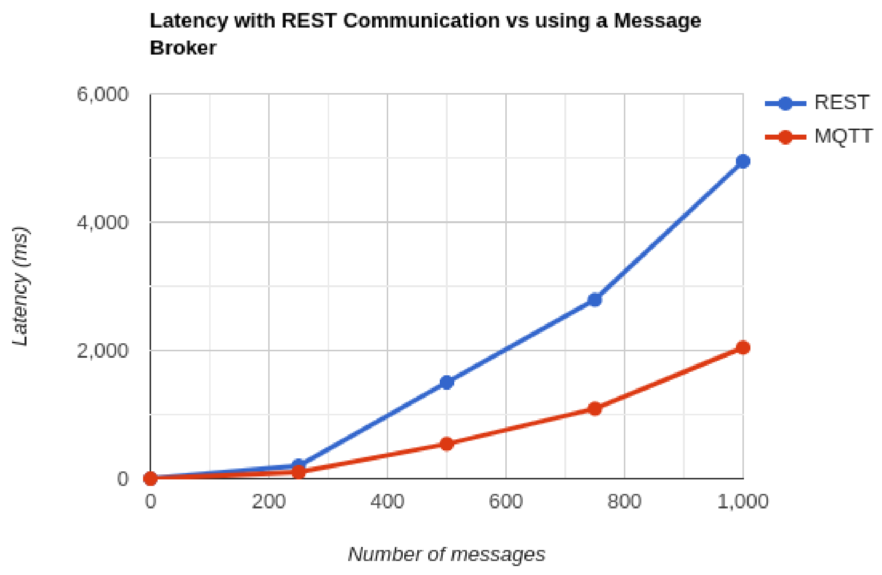


Figure 4.1: Latency of the system using a REST communication vs using a message Broker

It was evident that as the number of IoT devices connected to Edge X increased, the volume of messages grew over time, resulting in a significant increase in the number of received messages and subsequently, a rise in latency. Another challenge surfaced concerning the maximum capacity of the Edge X database: on average, each MQTT device was transmitting 27 messages per minute, with each message consisting of 200 bytes.

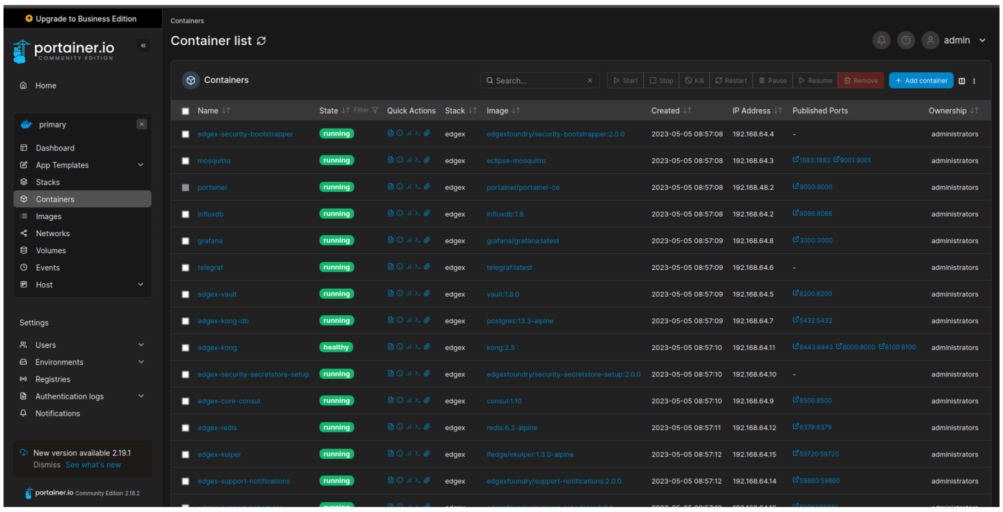
Time	Number of bytes
1 minute	5400
1 hour	324000
1 day	7776000

Table 4.1: Number of bytes to store the messages from one device

For 20 devices, the daily storage requirement amounted to 155,520,000 bytes (1.25 gigabytes), which proved to be both costly and memory-intensive for Edge X. The system's memory usage reached high levels after a few days, resulting in system slowdowns. This prompted the realization that a new external database was necessary to store the information before displaying it on the system's dashboard.

Both write and read times were increasing, causing further system performance issues. Implementing a rule to retain data on Edge X for a specific duration (in this case, one day) proved to be an optimal solution. This approach didn't slow down the system and allowed for the deletion of old data, making room for new data. With a one-day retention period, even with 20 devices continuously connected for 24 hours, the system operated more efficiently, and memory usage remained below the initial 90% threshold. No data was lost because it had been preserved in InfluxDB.

4.1.1 Portainer


Container list

Containers

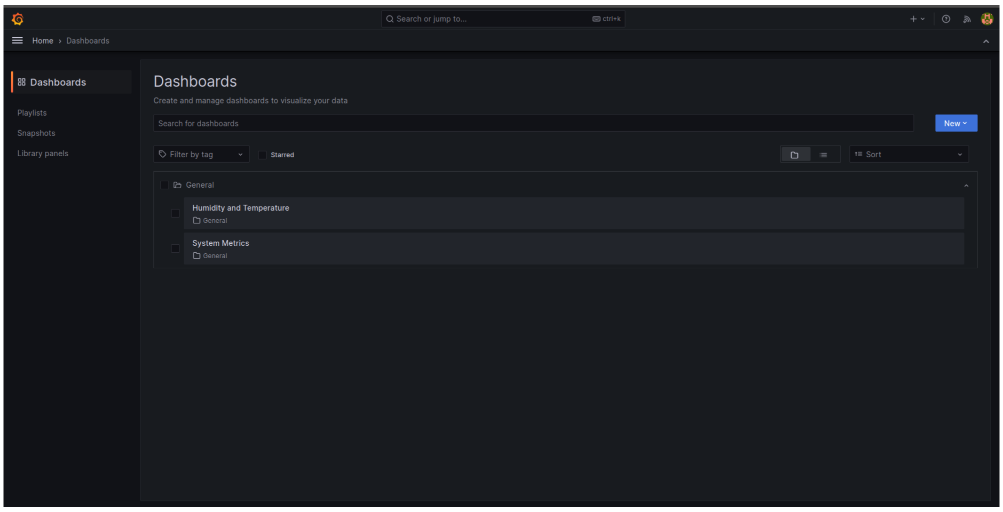
Name	State	Quick Actions	Stack	Image	Created	IP Address	Published Ports	Ownership
edgex-security-bootstrapper	running		edgex	edgexfoundry/security-bootstrapper:2.0.0	2023-05-05 08:57:08	192.168.64.4	-	administrators
mosquitto	running		edgex	eclipse-mosquitto	2023-05-05 08:57:08	192.168.64.3	1883:1883 9001:9001	administrators
portainer	running		edgex	portainer/portainer-ce	2023-05-05 08:57:08	192.168.48.2	9000:9000	administrators
influxdb	running		edgex	influxdb:1.8	2023-05-05 08:57:08	192.168.64.2	8086:8086	administrators
grafana	running		edgex	grafana/grafana:latest	2023-05-05 08:57:09	192.168.64.8	3000:3000	administrators
telegraf	running		edgex	telegraf:latest	2023-05-05 08:57:09	192.168.64.6	-	administrators
edgex-vault	running		edgex	vault:1.8.0	2023-05-05 08:57:09	192.168.64.5	8200:8200	administrators
edgex-kong-db	running		edgex	postgres:13.3-alpine	2023-05-05 08:57:09	192.168.64.7	5432:5432	administrators
edgex-kong	healthy		edgex	kong:2.5	2023-05-05 08:57:10	192.168.64.11	8443:8443 8000:8000 8100:8100	administrators
edgex-security-secretstore-setup	running		edgex	edgexfoundry/security-secretstore-setup:2.0.0	2023-05-05 08:57:10	192.168.64.10	-	administrators
edgex-core-consul	running		edgex	consul:1.10	2023-05-05 08:57:10	192.168.64.9	8500:8500	administrators
edgex-redis	running		edgex	redis:6.2-alpine	2023-05-05 08:57:11	192.168.64.12	6379:6379	administrators
edgex-kuiper	running		edgex	lfedge/ekuiper:1.3.0-alpine	2023-05-05 08:57:12	192.168.64.15	59720:59720	administrators
edgex-support-notifications	running		edgex	edgexfoundry/support-notifications:2.0.0	2023-05-05 08:57:12	192.168.64.14	59860:59860	administrators

Figure 4.2: Portainer.io with all the containers running

We also conducted an assessment of the EdgeX's edge-core-metadata container, which serves as the edge computing platform. During peak usage, it consumed 145.72 megabytes of memory. Within an hour, there was a smooth exchange of 80,000 bytes of data transmitted and 120,000 bytes received, with no transaction errors.

4.1.2 Grafana

The integration with Grafana was implemented as a means to visualize the logs, metrics, and values received by the system from the devices and Edge X components.


Dashboards

Create and manage dashboards to visualize your data

General
- Humidity and Temperature (General)
- System Metrics (General)

Figure 4.3: Grafana dashboard

It is feasible to observe real-time data from both sensors and edge metrics.

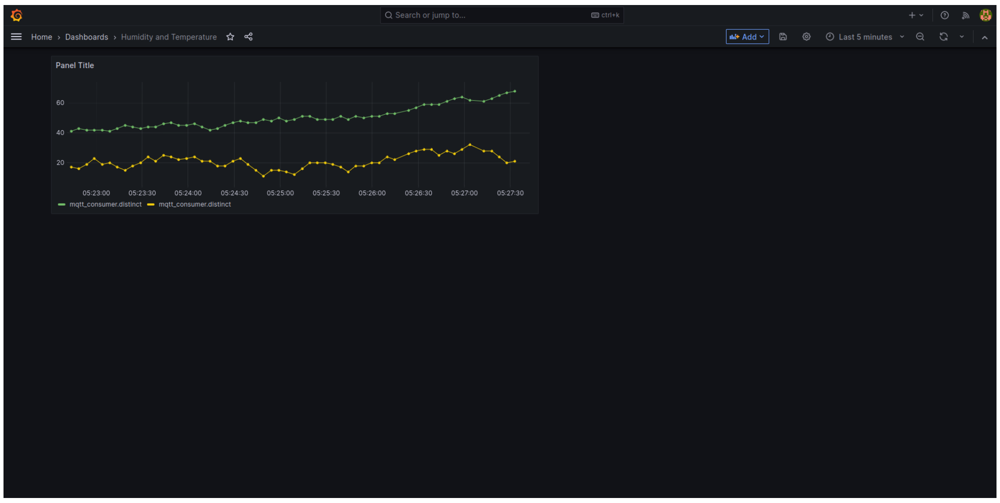


Figure 4.4: Humidity and Temperature values

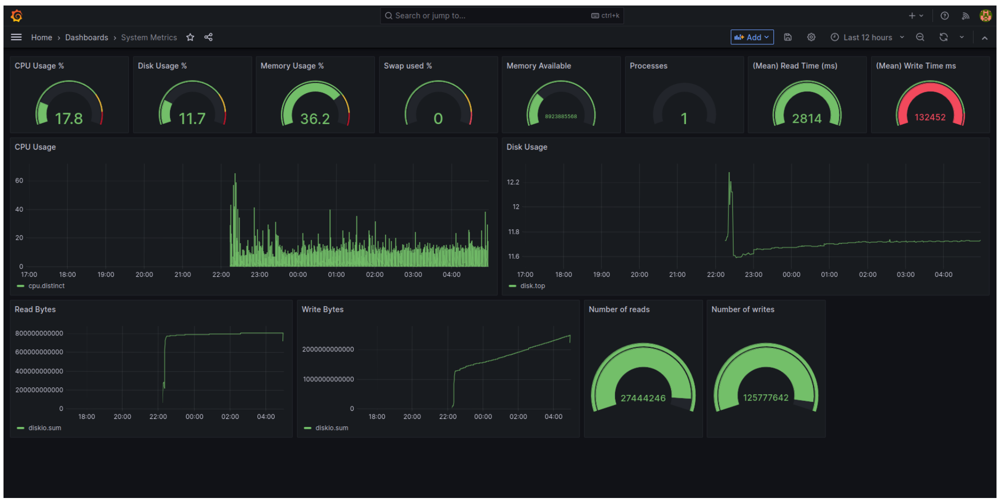


Figure 4.5: System metrics

The metrics analyzed where the CPU usage, Disk usage, swap usage, memory usage, memory available, number of concurrent process, read time, write time, read bytes, write bytes, number of reads of databases and number of writes on the databases.

Analysing the data, is possible to conclude that the CPU usage for each of these containers remained remarkably low, even when handling a substantial volume of data transmission in bytes per second. The data transmission process also proceeded without any reported errors. These findings demonstrate that our application can efficiently operate on systems with modest hardware specifications.

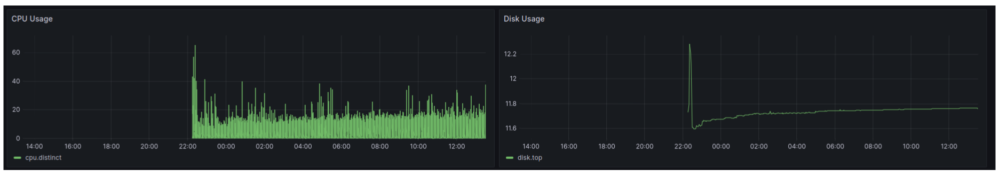
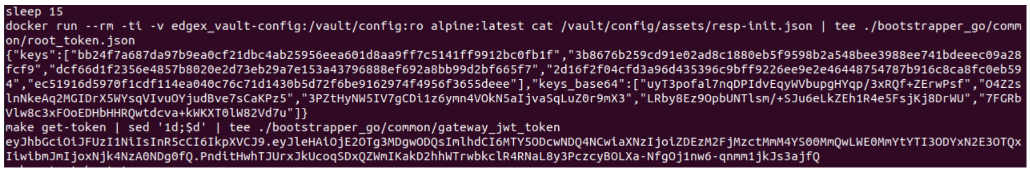


Figure 4.6: Cpu Usage (%) and Disk Usage (%)

To guarantee the security of transactions, all keys were securely stored using the Edge X vault, as mentioned earlier. This approach, when implemented, did not lead to notable service delays, as the keys were generated and stored when the containers started. While it did result in a slight increase in CPU usage, it was not a cause for concern as is possible to see in 4.6. The analysis indicates that the highest CPU and disk usage is observed during the system's reboot or startup phase, which is a predictable pattern. Subsequently, resource utilization returns to a standard operational state.



```
sleep 15
docker run --rm -ti -v edgex_vault-config:/vault/config:ro alpine:latest cat /vault/config/assets/resp-init.json | tee ./bootstrapper_go/comm
on/root_token.json
{"keys":["bb24f7a687da97b9ea0cf21dbc4ab25956eea601d8aa9ff7c5141ff9912bc0fb1f","3b8676b259cd91e02ad8c1880eb5f9598b2a548bee3988ee741bdeeec09a28
fcf9","dcf66d1f2356e4857b8020e2d73eb29a7e153a43796888ef692a8bb99d2bf665f7","2d16f2f04cfd3a96d435396c9bff9226ee9e2e46448754787b916c8ca8fc0eb59
4","ec51916d5970f1cdf114ea040c76c71d1430b5d72f6be9162974f4956f3655deee"],"keys_base64":["uyT3pofal7nqDPIdvEqyWVbupgHYqp/3xRQf+ZErwPsf","O4Z2s
lnNkeAq2MGIDrX5WYsqVIvuOYjudBve7sCaKPz5","3PZtHyNW5IV7gCDi1z6ymn4VOkN5aIjvaSqLuZ0r9mX3","LRby8Ez9OpbUNTlsm/+SJu6eLkZEh1R4e5FsjKj8DrWU","7FGRb
Vlw8c3xFOoEDHbHHRQwtdcva+kWKXT0lW82Vd7u"]}
make get-token | sed '1d;$d' | tee ./bootstrapper_go/common/gateway_jwt_token
eyJhbGciOiJFUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE2OTg3MDgwODQsImlhdCI6MTY5ODcwNDQ4NCwiaXNzIjoiZDEzM2FjMzctMmM4YS00MmQwLWE0MmYtYTI3ODYxN2E3OTQx
IiwibmJmIjoxNjk4NzA0NDg0fQ.PnditHwhTJUrxJkUcoqSDxQZWmIKakD2hhWTrwbkclR4RNaL8y3PczcyBOLXa-NfgOj1nw6-qnmm1jkJs3ajfQ
```

Figure 4.7: Security keys generated to ensure that all the devices are supposed to comunnicate with Edge X

Memory usage initially increases but eventually stabilizes and remains constant as new values are deleted and news are written. Both the available memory and the bytes read and written maintain a steady state.

Regarding data processing, the expected pattern holds true with the average read time being shorter than the write time. It takes roughly 2 seconds to read all values, while writing these values to the InfluxDB database requires approximately 13.2 seconds.

At times, the IoT devices would encounter issues and it was crucial to have a backup device on standby when one became unavailable. This was achieved through the use of alerts triggered when Edge X ceased receiving data from the source in the last 5 seconds (the time it takes for the UI to refresh the UI dashboard). When this occurred, Edge X attempted to send a signal to establish communication with the device. If this attempt failed, it would activate the second IoT device according to a predefined rule, and this device would take over until the original one became available again.

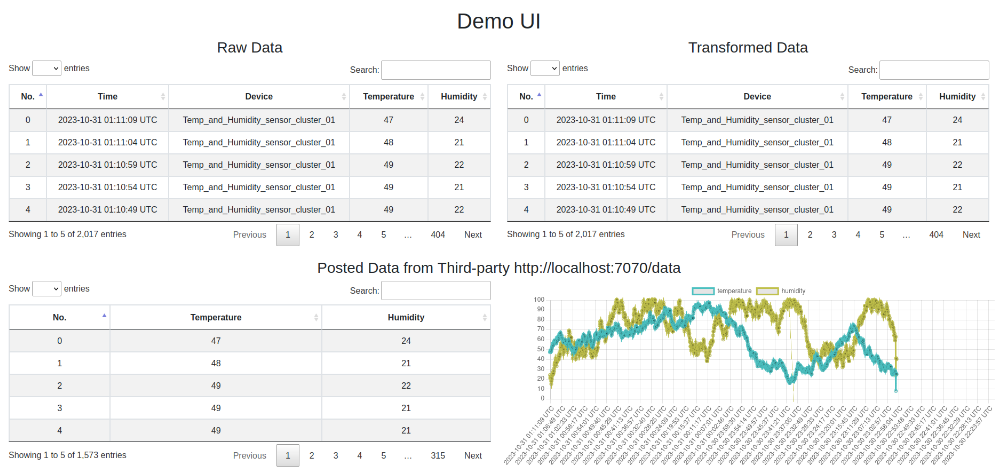
Both IoT devices were configured with identical rules, ensuring that the transition was seamless and didn't create the impression of data originating from different sources while the primary device was temporarily unavailable, using the same tags on the Edge X.

4.2 UI with all the information

Ultimately, we achieved the capability to replicate another system and establish communication between both using a cloud-based load balancer to distribute network traffic when the dashboards experienced significant latency. This arrangement allowed access to the dashboards of both systems, and Edge X could effectively communicate with both.

In the event that one Edge X instance became unavailable, the system would attempt to locate another one based on a predefined rule (hierarchically). The IoT device would then establish a connection with the available Edge X instance until the primary one, according to the rule, became accessible once more.

4.2.1 Demo



Demo UI

Raw Data

Show entries Search:

No.	Time	Device	Temperature	Humidity
0	2023-10-31 01:11:09 UTC	Temp_and_Humidity_sensor_cluster_01	47	24
1	2023-10-31 01:11:04 UTC	Temp_and_Humidity_sensor_cluster_01	48	21
2	2023-10-31 01:10:59 UTC	Temp_and_Humidity_sensor_cluster_01	49	22
3	2023-10-31 01:10:54 UTC	Temp_and_Humidity_sensor_cluster_01	49	21
4	2023-10-31 01:10:49 UTC	Temp_and_Humidity_sensor_cluster_01	49	22

Showing 1 to 5 of 2,017 entries Previous 1 2 3 4 5 … 404 Next

Transformed Data

Show entries Search:

No.	Time	Device	Temperature	Humidity
0	2023-10-31 01:11:09 UTC	Temp_and_Humidity_sensor_cluster_01	47	24
1	2023-10-31 01:11:04 UTC	Temp_and_Humidity_sensor_cluster_01	48	21
2	2023-10-31 01:10:59 UTC	Temp_and_Humidity_sensor_cluster_01	49	22
3	2023-10-31 01:10:54 UTC	Temp_and_Humidity_sensor_cluster_01	49	21
4	2023-10-31 01:10:49 UTC	Temp_and_Humidity_sensor_cluster_01	49	22

Showing 1 to 5 of 2,017 entries Previous 1 2 3 4 5 … 404 Next

Posted Data from Third-party http://localhost:7070/data

Show entries Search:

No.	Temperature	Humidity
0	47	24
1	48	21
2	49	22
3	49	21
4	49	21

Showing 1 to 5 of 1,573 entries Previous 1 2 3 4 5 … 315 Next

Figure 4.8: Demo UI with all the Edge X available to analyze

The dashboard undergoes automatic refreshing at 5-second intervals, ensuring the most real-time presentation of values. It's worth noting that in the data list, the value at index 0 always corresponds to the most recent data point.

The system supports comprehensive analysis, enabling access to both raw data (the message as received before any processing) and transformed data (the message after consumption and processing). Moreover, the system offers the functionality to search and apply filters to the data based on individual devices, facilitating precise data retrieval and examination.

When using both Edge X, the latency of the information that arrives at the dashboard that are connected to the designed Edge don't increase to much. The problem is when the other Edge X tries to access the other devices to demonstrate in the other dashboard and the workload is significant.

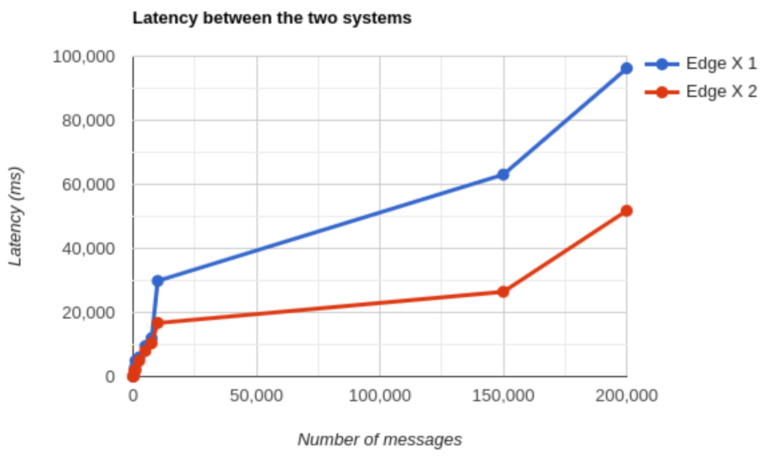


Figure 4.9: Latency between both Edge X's

CHAPTER 5

Discussion and Conclusion

This dissertation sought to address the absence of a market-ready architecture by developing an edge computing solution that expands its applicability across a wider range of scenarios.

In conclusion, the concept of a computing continuum within a federated network fabric represents a promising and transformative approach to addressing the complex and evolving needs of modern networked systems. This dissertation has explored the fundamental principles, design considerations, and potential benefits of integrating computing resources seamlessly across a distributed and edge federated network fabric. Through an examination of the key components, challenges, and emerging technologies associated with this paradigm, several important insights can be drawn.

First and foremost, it is evident that the demand for computing resources is expanding rapidly, fueled by the proliferation of data-intensive applications, the IoT, and the advent of edge computing. Traditional, centralized data centers are increasingly ill-suited to meet the latency and bandwidth requirements of these applications. A federated network fabric, which integrates computing resources at the network edge and seamlessly coordinates their use, offers a compelling solution to these challenges.

Moreover, this dissertation has shed light on the vital role of SDN and NFV in enabling the flexible allocation of resources within a federated network fabric. By decoupling network functions from dedicated hardware and centralizing their management through software, these technologies empower network operators to respond dynamically to changing demands and optimize resource utilization.

The concept of a computing continuum has also been demonstrated as a powerful approach for achieving enhanced user experiences, reduced latency, and improved fault tolerance. By moving computing resources closer to the data source or user, applications can respond faster, increasing overall system efficiency and reliability.

However, it is crucial to acknowledge the challenges and complexities associated with implementing a federated network fabric and computing continuum. Security and privacy concerns, interoperability issues, and the need for standardization remain significant hurdles that must be addressed by industry stakeholders and policymakers.

As the world of networking continues to evolve, it is evident that the paradigm of a computing continuum within a federated network fabric holds great potential for meeting the growing demands of modern applications and services.

Initial testing with the Edge X highlighted the imperative to reduce the volume of continuously stored data to align with the network system's requirements. The introduction of a new edge-independent database significantly enhanced system efficiency and bolstered data security through redundancy measures. Incorporating various rules and alerts further enhanced system redundancy, ensuring prolonged system uptime. Additionally, the implementation of a new communication protocol not only bolstered security but also expedited response times, mitigating issues related to failures and data gaps at the edge.

The results obtained from this work confirm that the implemented changes are not only well-received but also seamlessly compatible with the system, contributing substantially to its overall enhancement.

5.1 Future work

For future work in the project, several key areas of investment and development have been identified to further enhance the Edge X architecture and its overall performance, such as continue to improve and optimize the Edge X rules and architecture of the system to reduce latency and become the system more robust and efficient, invest in the communication between different systems and continue to investigate the results. It would be beneficial to make a trace of the alerts triggered on the UI. Furthermore, significant efforts should be dedicated to refining rules with the incorporation of AI for decision-making, going beyond mere alerts - as an illustration, if the sensor registers an excessively high temperature, the system should activate the air conditioner, and if the temperature does not subsequently decrease, the system should initiate the shutdown of that device.

Incorporating these key focus areas into future work will not only bolster the capabilities of the project but also position it to adapt and thrive in an increasingly dynamic and demanding technological landscape. By striving for lower latency, improved communication, AI-driven decision-making, and overall system robustness and efficiency, the project can deliver more reliable, responsive, and intelligent solutions that meet the evolving needs of modern networked environments.

References

[1] Z. Zhu, G. Han, G. Jia, and L. Shu, “Modified densenet for automatic fabric defect detection with edge computing for minimizing latency,” *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9623–9636, 2020. DOI: 10.1109/JIOT.2020.2983050.

[2] D. Balouek-Thomert, E. G. Renart, A. R. Zamani, A. Simonet, and M. Parashar, “Towards a computing continuum: Enabling edge-to-cloud integration for data-driven workflows,” *The International Journal of High Performance Computing Applications*, vol. 33, no. 6, pp. 1159–1174, 2019. DOI: 10.1177/1094342019877383. eprint: https://doi.org/10.1177/1094342019877383. [Online]. Available: https://doi.org/10.1177/1094342019877383.

[3] L. M. Contreras and C. J. Bernardos, “Overview of architectural alternatives for the integration of etsi mec environments from different administrative domains,” *Electronics*, vol. 9, no. 9, 2020, ISSN: 2079-9292. DOI: 10.3390/electronics9091392. [Online]. Available: https://www.mdpi.com/2079-9292/9/9/1392.

[4] M. L. F. Sindjoung, M. Velempini, and A. B. Bomgni, “A mec architecture for a better quality of service in an autonomous vehicular network,” *Computer Networks*, vol. 219, p. 109 454, 2022, ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2022.109454. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128622004881.

[5] S. Ranganath, “Chapter three - industry initiatives across edge computing,” in *Edge/Fog Computing Paradigm: The Concept Platforms and Applications*, ser. Advances in Computers, P. Raj, K. Saini, and C. Surianarayanan, Eds., vol. 127, Elsevier, 2022, pp. 63–115. DOI: https://doi.org/10.1016/bs.adcom.2022.03.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0065245822000468.

[6] C.-Y. Weng, C.-T. Li, C.-L. Chen, C.-C. Lee, and Y.-Y. Deng, “A lightweight anonymous authentication and secure communication scheme for fog computing services,” *IEEE Access*, vol. 9, pp. 145 522–145 537, 2021. DOI: 10.1109/access.2021.3123234. [Online]. Available: https://doi.org/10.1109/access.2021.3123234.

[7] D. Rosendo, P. Silva, M. Simonin, A. Costan, and G. Antoniu, “E2Clab: Exploring the Computing Continuum through Repeatable, Replicable and Reproducible Edge-to-Cloud Experiments,” in *Cluster 2020 - IEEE International Conference on Cluster Computing*, Kobe, Japan, Sep. 2020, pp. 1–11. DOI: 10.1109/CLUSTER49012.2020.00028. [Online]. Available: https://hal.science/hal-02916032.

[8] N. Hassan, K.-L. A. Yau, and C. Wu, “Edge computing in 5g: A review,” *IEEE Access*, vol. 7, pp. 127 276–127 289, 2019. DOI: 10.1109/access.2019.2938534. [Online]. Available: https://doi.org/10.1109/access.2019.2938534.

[9] Y. Cheng, H. Zhang, and Y. Huang, “Overview of communication protocols in internet of things: Architecture, development and future trends,” in *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, 2018, pp. 627–630. DOI: 10.1109/WI.2018.00-25.

[10] R. N. S. Jyothi, J. Sireesha, A. Mahitha, B. Ruchitha, and E. Deepthi, “Protection and saving of delicate data by using cloud computing,” in *2022 International Conference on Electronics and Renewable Systems (ICEARS)*, 2022, pp. 1660–1667. DOI: 10.1109/ICEARS53579.2022.9752329.

[11] T. T. Huynh, T. D. Nguyen, T. Hoang, L. Tran, and D. Choi, “A reliability guaranteed solution for data storing and sharing,” *IEEE Access*, vol. 9, pp. 108 318–108 328, 2021. DOI: 10.1109/ACCESS.2021.3100707.

[12] I. Sittón-Candanedo, "GECA: A global edge computing architecture," in *The role of Artificial Intelligence and distributed computing in IoT applications*, Ediciones Universidad de Salmanca, Sep. 2020, pp. 85–96. DOI: `10.14201/0aq02878596`. [Online]. Available: `https://doi.org/10.14201/0aq02878596`.

[13] M. Caprolu, R. Di Pietro, F. Lombardi, and S. Raponi, "Edge computing perspectives: Architectures, technologies, and open security issues," in *2019 IEEE International Conference on Edge Computing (EDGE)*, 2019, pp. 116–123. DOI: `10.1109/EDGE.2019.00035`.