International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Review on Maintainable, Efficient & Reliable IoT Middleware for Critical Applications

Ms. Kirti Singh [1], Ms. Aditi Dubey [2], Ms. Sakshi Tale [3], Mr. Om Gavhane [4], Dr. Yashwant Dongre [5]

Submitted: 07/02/2024 **Revised**: 10/03/2024 **Accepted**: 20/03/2024

Abstract: The necessity of the IoT Middleware in critical applications cannot be overstated. Middleware plays a pivotal role in military operations, healthcare delivery, robotics, and farming, where seamless integration of IoT devices is crucial. As a conduit between devices and applications, it ensures real-time data transmission, enhances security protocols and facilitates interoperability across heterogeneous networks and platforms. This functionality is paramount in critical settings, where timely and accurate information can indicate the difference between success and failure. Through the utilization of IoT Middleware, these applications achieve heightened situational awareness, precise data analytics, and context-aware communication, thereby revolutionizing operational strategies and decision-making processes. Research conducted on IoT Middleware has led to remarkable achievements, including improved scalability, enhanced security measures, and seamless integration with existing infrastructure. As a result, critical applications across diverse domains have witnessed significant advancements in operational efficiency, resource optimization, and overall effectiveness, ultimately leading to better outcomes in military missions, patient care delivery, robotic automation, and precision agriculture.

Keywords: *Scalability, Reliability, Maintainability, Interoperability, Context Awareness*

1. Introduction

1.1. Overview

Our interactions with the physical world have been completely transformed by the Internet of Things (IoT). By utilizing a network of sensors and devices, real-time data collection and analysis is made possible, resulting in advancements across multiple industries. IoT middleware is a robust software layer to manage this complex data flow and communication. Within an IoT system, IoT middleware serves as a link between resource-constrained devices and applications.[1] It makes interacting and communicating

across various platforms, apps, and gadgets easier. A unified IoT ecosystem is produced by middleware, which offers standardized APIs, data translation, security protocols, and device management. This increases IoT deployments' scalability and efficacy and facilitates the process of gaining insightful information from various data sources.

1.2. Motivation

A critical gap has been revealed by the growing use of IoT devices in vital applications such as robotics, military operations, healthcare, and agriculture. This gap is the need

for reliable and adaptable IoT middleware. Real-time data exchange and the smooth integration of various devices are essential to these applications. However, current middleware solutions frequently lack the scalability, security, and interoperability needed for optimal performance in such demanding environments.[4]

This motivates the development of next-generation IoT middleware research. The goal of the research is to enable critical applications through improved scalability, enhanced security features, and seamless integration with current infrastructure. Context-aware communication, data analytics, and situational awareness will all advance as a result of this. In the long run, these studies can completely transform these industries' operational plans and decision-making procedures, resulting in major gains in productivity, resource optimization, and general efficacy.[38]

1.3. Problem Definition and Objectives

1.3.1. Problem Definition

An important limitation has been made clear by the increasing use of IoT devices in vital industries like healthcare, robotics, agriculture, and the military: the available IoT middleware solutions are insufficient.[27] Real-time data processing and seamless device-to-device communication are essential for these applications. Unfortunately, scalability, security, and interoperability issues with current middleware prevent it from performing at its best in these demanding environments.[3] To overcome these limitations and enable vital applications to achieve greater situational awareness, data analysis, and communication, research on enhanced IoT middleware is required. This will eventually lead to advancements in efficiency, resource optimization, and overall effectiveness.

[1, 3, 4, 5] *Department of Computer Engineering Vishwakarma Institute of Information Technology, Pune, India.* [1] *singhkirti13931@gmail.com,* [3] *talesakshi64@gmail.com,* [4] *om2002gavhane@gmail.com,* [5] *yashwant.dongre@viit.ac.in*
[2] *Department of Information Technology Vishwakarma Institute of Information Technology, Pune, India. aditidubey.ad.6@gmail.com*

1.3.2. Objectives

The objectives determined by Middleware Properties and Critical Applications:

[1] Improved Data Integrity and Security for Healthcare:

Provide strong security measures in the middleware to protect private medical information that is transferred from implants and wearables.[2] Put data integrity first by using techniques for validation and error correction to make sure that medical professionals receive accurate patient information.[5]

[2] Military Situational Awareness and Real-Time Communication:

Improve the middleware to transfer data with minimal latency so that soldiers, drones, and command centers can communicate in real-time. Provide middleware functionality that makes coordinated and transparent communication possible during crucial operations.[31]

[3] Agriculture Data Management That Is Both Scalable and Effective:

Provide middleware solutions that are scalable to manage the massive volumes of data gathered from agricultural sensor networks spread out geographically. Use effective data aggregation and filtering strategies at the edge to reduce network traffic and maximize resource use.[30]

[4] Smooth Data Flow and Integration for Smart Cities:

Create middleware that enables smooth data flow throughout the entire infrastructure by integrating with a variety of sensor networks and city management systems. Enhance the middleware's data processing to allow for real-time analytics and quick decision-making for smart city resource allocation.[36]

1.4. Scope and Limitations

1.4.1. Scope

The purpose of this research is to overcome the shortcomings in the current IoT middleware that prevent it from being used effectively in vital applications such as agriculture, healthcare, and the military. The range includes:

[1] Scalability and real-time processing: Creating middleware techniques that minimize bottlenecks and delays while managing the high volume and velocity of data generated by vital applications.[17]

[2] Enhanced security: It involves looking into ways to fortify middleware security protocols and protect sensitive data from alterations or breaches.[19]

[3] Devices with limited resources: Creating middleware solutions that are lightweight and effectively function on devices with constrained processing and battery life.

[4] Interoperability and standardization: Investigating ways to make various middleware solutions more compatible with one another as well as creating common data formats and communication protocols to enable smooth integration with important applications.[39]

[5] Context-aware decision support: It refers to the process of incorporating middleware with context-awareness features that enable data interpretation based on situational factors and offer real-time decision support for better response and action in emergencies.[38]

1.4.2. Limitations

Like any other research, this research could have some restrictions that should be taken into account. The following are some possible limitations of the research:

[1] Concentration on particular applications: The research may give priority to advancements for particular, vital applications, such as the military or the healthcare system, which could restrict the applicability of the findings to other, equally important areas, such as smart cities or agriculture.

[2] Limited scope of addressed limitations: Although the research addresses issues related to resource constraints, interoperability, security, scalability, and context awareness, it may not address other emerging issues related to privacy, energy efficiency, or regulatory compliance in IoT middleware.

[3] Real-world Implementation vs. Prototyping: Research may concentrate on creating and evaluating solutions in a safe setting. Unexpected difficulties may arise when putting these solutions into practice and assessing them in the dynamic, complex settings of important real-world applications.[18]

[4] Scalability of Solutions: The study may offer remedies for problems that arise from the deployment of a single, crucial application.[23] It may be necessary to conduct additional research and make adjustments to scale these solutions to manage the enormous and intricate networks of numerous crucial applications.

2. Literature Survey

[1] Adib Mehedi, Adib Hassan Tokee, Surovi Islam, and Md Saef Ullah Miah (2020) studied the potential of Middleware in the Healthcare Sector for improving patient care delivery and expediting healthcare procedures through both its functionality and design. The Algorithms used in this study were patient-ID generation, data exchange, and emergency communication. Future research can be performed on bolstering security protocols, resolving interoperability issues, incorporating real-time data analytics capabilities, and improving user experience and usability.

[2] Sunitha Lingam (2021) investigated how IoT makes medication adherence, disease management, predictive healthcare, and widespread monitoring possible. For this research, algorithms such as data transmission protocols, Data Analysis Algorithms (regression, classification, and clustering), Data Mining techniques, Forecast-Based Alerting Systems, Systems for Reminding Medication & Techniques for Remote Monitoring were used. Future research should emphasize real-world applications and IoT validation in the healthcare industry to achieve observable advancements in patient care and healthcare delivery.

[3] Ghofrane Feresi (2020) focused on middleware solutions specifically designed to make it easier to integrate new devices into the IoHT network and implement healthcare applications. The methodologies used for this research were Techniques for Middleware Analysis, Methods for Developing a Taxonomy (expert consensus, feature extraction, and hierarchical clustering), and Evaluation Metrics & Methods for Comparative Analysis. Future research should emphasize validating IoHT middleware solutions and incorporating them into real-world applications to achieve noticeable gains in healthcare quality, efficiency, and accessibility.

[4] Venki Balasubramanian and Jolfaei (2021) drew attention to how IoMT may help improve patient outcomes and reduce healthcare costs, especially with the advent of Healthcare Monitoring Applications (HMA). The algorithms used were Techniques for Remote Patient Monitoring, Access Techniques for Electronic Medical Records, and Communication Techniques. The creation of an Assistive Care Loop Framework (ACLF), intended for real-time monitoring of expectant mothers, serves as an example of the general framework for creating HMAs proposed in this paper.

[5] Rita Zgheib, Emmanuel Conchon, and Rémi Bastide (2019) examined interoperability issues and potential solutions in Internet of Things (IoT) contexts, with a focus on the healthcare industry. It discusses the variety of connected devices and data formats and offers a summary of the technical and semantic ways to solve these problems. The mechanisms used are the Synopsis of the Solutions and Solutions for Semantic Middleware. This study also emphasizes the significance of semantic middleware solutions for achieving full interoperability.

[6] Madhavi Latha Challa, K. L. S. Soujanya, and C. D. Amulya (2020) aim to improve the security and efficiency of IoT medical devices for remote patient monitoring. The approaches used in this research are The Internet of Health Systems Security and Interoperability (IHSI) Method & Security Inputs. Subsequent investigations may concentrate on enhancing the modules and algorithms of the IHSI approach to enhance its functionality and flexibility in response to changing healthcare requirements and technological breakthroughs.

[7] R. Venkateswara Reddy, D. Murali, and J. Rajeshwar (2019) highlighted the importance of contextual middleware for IoT applications, emphasizing its adaptability and user-centered design. The algorithms used were Security Enhancement Techniques, Context-Awareness Techniques, Methods for Visualizing IoT Data & Big Data Analysis Algorithms. This study emphasizes the value of improving security, privacy, and data visualization through cloud-based Big Data analysis while highlighting current IoT middleware techniques.

[8] Manisha Banka, Ankita Mishra, and Sushreeta Tripathy (2022) examine trends, applications, and typical difficulties in putting smart healthcare solutions into practice. It also addresses the opportunities and challenges in providing effective healthcare facilities, particularly for the elderly. The mechanisms used include IoT Integration, Data Analytics, Telemedicine Solutions, Resource Optimization & Continuous Monitoring. Future research should concentrate on resolving common issues with IoT implementation in healthcare settings, such as data security, interoperability, and regulatory compliance.

[9] F. Li, L. Chen, Y. Wang, and X. Wang (2022) studied the status of the global military IoT, presented the military application scenarios of IoT OS, and summarized the technical requirements of IoT OS for military applications. The proposed framework involves optimizing the architecture design, enhancing standards, and integrating new technologies to develop a Military IoT OS. This study analyzes the military application scenarios of IoT OS from three dimensions: peacetime, wartime, emergency, military operations, and non-war military operations.

[10] Rune Langleite, Carsten Griwodz, and Frank T. Johnsen (2021) explore the development of a prototype Military IoT (MIoT) soldier wearable using commercially available software and hardware. The prototype uses a private network constructed from free open-source software and communicates using a low-power long-range wide area network (LoRaWAN), independent of existing infrastructure. This study investigates the applicability of an MIoT subsystem in the form of a soldier wearable device, based on the primary goal of using IoT to improve combat effectiveness through enhanced Situational Awareness (SA).

[11] Dey, E., Walczak, M., Anwar, M. S., & Roy, N.(2023) proposed the tested middleware for using Gazebo as a physics-based ROS simulator and NS-3 as a wireless network simulator, with comparisons made against real-time wireless network simulator EMANE and master-based ROS1 synchronization middleware. The experimental results demonstrate that the proposed middleware outperforms traditional ROS1 systems and real-time

network simulators in terms of reducing packet loss and improving transmission rates, particularly in challenging non-line-of-sight environments with heterogeneous agents.

[12] Adib Mehedi, Adib Hassan Tokee, Surovi Islam, and Md Saef Ullah Miah (2020) studied the potential of Middleware in the Healthcare Sector for improving patient care delivery and expediting healthcare procedures through both its functionality and design. The Algorithms they used in this study are Patient-ID Generation, Data Exchange, & Emergency Communication. We can perform Future research on Bolstering security protocols, resolving interoperability issues, incorporating real-time data analytics capabilities, and improving user experience and usability.

[13] M. T. Moghaddam, É. Rutten and Giraud (2020) proposed a process for performing a methodological literature review on adaptive middleware support for cyber-physical systems (CPS) and the Internet of Things (IoT). The algorithms they used were the Classification and Extraction Framework, Criteria-based Selection, Screening and Evaluation, and Utilization of Academic Databases and Industrial use cases. The results are expected to contribute to a deeper understanding of adaptive middleware support for IoT and CPS, potentially leading to changes in practices, methodologies, and future research directions within the domain. It will support the improvement of user-centered design, human-system interaction, interoperability and standardization, machine learning and artificial intelligence integration, and evaluation of implementation techniques.

[14] Cédric Melançon, Guillaume Simard, Maarouf Saad, Kuljeet Kaur, and Julien Gascon-Samson (2023) demonstrated 'BlazeFlow', their multilayer data flow solution based on publish-subscribe abstractions that can move data to and from any layer of the cloud-to-device continuum. As evidenced by the evaluation in an autonomous robotics use case, it manages many protocols, enables data transit across cloud-to-device layers, and supports bandwidth-intensive and time-critical applications. It supports bandwidth-intensive and time-critical services deployed at various layers of the system. This illustrates BlazeFlow's ability to tolerate multilayer data lows at a high frequency and low latency by evaluating an early design over an autonomous robotics use case.

[15] M. Bazzani and Davide Conzon (2021) studied how an approach that utilizes the VIRTUS IoT middleware offers a strong substitute for the majority of the present IoT solutions, which are SOA (Service Oriented Architecture) based. The algorithms used were the Instant Messaging Protocol (XMPP), Remote Body Movement Monitoring System, and Modular Architecture Deployment. VIRTUS IoT middleware enhances e-health solutions with real-time communication and remote monitoring capabilities. It provides a gateway to guarantee interoperability, adherence to healthcare standards, scalability, security, and integration with developing technologies.

[16] R.A.K. Lakpriya, W.A.S. Rathsara, Piumi Fernando and H.S. Thenuwara, L.O Ruggahakotuwa, and A. Senarathne (2022) suggested an IoT device security middleware that would gather data produced by IoT devices, check them for malicious activities, and then sound an alarm if a threat is found within the IoT network. The proposed SDN architecture, which combines VPN, cloud computing, and fog computing technologies to build a safe, adaptable, quick, and flexible network architecture, incorporates a secure middleware. It considerably enhances the security posture of IoT networks by effectively identifying and addressing security risks in real-time while maintaining peak performance and efficient use of available resources.

[17] G. Bhandari, Andreas Lyth, Andrii Shalaginov, and Tor-Morten Grønli (2023) suggested a methodology that includes a framework for identifying malware attacks through the application of artificial intelligence (AI) techniques in a variety of dispersed and varied circumstances. This entails using artificial intelligence techniques, specifically DNN models, to analyze network traffic data in real-time to find malware and assaults in IoT ecosystems, thereby improving the security of Smart Environments. Consequently, AI uses DNNs to accurately identify multilevel assaults in Smart Environments. It minimizes the impact on resources while they are being deployed and efficiently detects contaminated Internet of Things devices, thereby saving money and labor.

[18] Eryk Schiller, Elfat Esati, B. Stiller (2021) suggested the endeavour that puts into practice and assesses face recognition-based access control. Combining these algorithms makes it easier to create a facial recognition-based, all-inclusive access management system that uses blockchain, the IoT, and AI to operate securely and efficiently. They efficiently developed a system that delivers exceptional qualities in terms of image quality, end-to-end delay, and energy efficiency, and leveraged blockchains for immutable permanent storage, the Internet of Things for video surveillance, and Artificial Intelligence for face identification.

[19] Qamar Alfalouji, Thomas Schranz, A. Kümpel, M. Schraven, T. Storek, Stephan Gross, A. Monti, D. Müller, G. Schweiger (2022) analyzed IoT middleware platforms for energy systems should offer a range of tools and features that can be supported by any future effective, adaptable, and interoperable IoT middleware while taking market demands into account. The algorithms used were Literature Review and Expert Survey Methodology, IoT Middleware Functionality Analysis, Security and Privacy Algorithms, and Energy Optimization Algorithms. IoT middleware is mostly used by experts for monitoring and visualizing energy usage, and energy optimization is a

future goal. Nonfunctional needs, such as privacy and security, are critical; energy systems require adaptable and interoperable IoT middleware.

[20] Shalmoly Mondal, A. Hassani, P. Jayaraman, P. D. Haghighi, and Dimitrios Georgakopoulos (2021) suggested ARDG-IoT, a framework for modeling the requirements of IoT applications and enabling data generation. The framework makes it possible to benchmark IoT middleware solutions consistently, making it easier to compare and evaluate well-informed decision-making. The algorithms used are the IoTSySML Model Creation, IoT Data Generation Algorithm, and Evaluation Algorithm where IoT application requirements are formally captured by IoTSySML. Subsequently, IoT data were generated for benchmarking using the IoT data simulator. Uniform benchmarking makes it possible to select the best middleware platform for a variety of IoT applications.

[21] Guru Prasad Bhandari, Andreas Lyth, Andrii Shalaginov, and Tor-Morten Groenli (2022) presented a new paradigm and middleware based on Artificial Intelligence (AI) for detecting threats in flexible Smart Environments. The approach they used involved combining multilevel network traffic data from several IoT devices, called data aggregation. The algorithms used were the AI Model Application, Efficiency Evaluation, Deployment, and Testing. The test findings show that models and middleware powered by artificial intelligence can successfully identify cyberattacks in a variety of Smart Environments. The suggested four-step procedure demonstrates the effectiveness of AI techniques in improving IoT cybersecurity by enabling data-driven multi-agent malware and attack detection.

[22] Amrita Rai, Deepti Sharma, Shubhyansh Rai, Amandeep Singh, Krishna Kant Singh (2021) discussed the recently developed area of IoT-aided robotics, which aims to improve the capabilities of both systems by fusing IoT and robotics technologies. It draws attention to the concept of the "Internet of Robotic Things (IoRT)" and its possible uses in several industries, including agriculture, medicine, defense, industrial plants, and rescue operations. The algorithms used were Intelligent Robotics Structures, the Internet of Robotic Things (IoRT), and Semantic-Oriented Approaches. IoT-aided robotics presents numerous potential problems in various domains, such as distributed computing, network security, consensus methods, and communication networks. For IoRT systems to advance and be widely used in various settings, including rescue operations, farming, medicine, defense, and industrial sectors, these obstacles must be overcome.

[23] Lal Verda Çakır, Tuğçe Bilen, Mehmet Özdem, and Berk Canberk (2023) suggested DT Middleware makes effective DT-based farming possible by enabling context-aware communication, lowering resource usage in IoT devices, and reducing latency and round-trip time (RTT). The algorithms used were YANG-powered DT Middleware, DT-specific Aggregation Scheme, and Context-aware Communication.

[24] A. Steven, Suharnawi, Filmada Ocky Saputra, Rama Aria Megantara, F. Alzami, and P. Andono (2023) explained how by using machine learning algorithms to process the collected data, farmers can obtain precise weather forecasts that help them make well-informed decisions about when to sow and harvest their crops—important steps that are necessary for successful crop management and increased yields. The algorithms used were Machine Learning Techniques, Data Analytics Algorithms, Middleware Architectures and APIs. Increased yields and previously unattainable efficiency have been provided to farmers through the application of IoT, AI, and machine learning in agriculture. Farmers may increase the planting and harvesting schedule efficiency and profitability by utilizing real-time data and analytics, which will benefit their operations.

3. Design & Working of IoT Middleware

3.1. Architecture of IoT Middleware

IoT middleware architecture can differ based on the vendor and implementation, but the following describes a typical layered approach:

[1] Perception Layer (Device Layer):

This layer, which consists of all the sensors, actuators, and other Internet of Things devices gathering and possibly modifying real-world data, serves as the structural basis.[22]

Different protocols, such as Bluetooth, Wi-Fi, cellular networks, or specialized IoT protocols (like LoRaWAN), are used by devices to communicate with one another.[24]

[2] Network Layer (Transport Layer):

This layer controls data transfer between devices and the middleware. It guarantees dependable data transfer and specifies the communication protocols used, such as TCP/IP, MQTT, and CoAP.[16]

In this layer, gateways may compile data from various devices and send it to the middleware.[20]

[3] IoT Middleware Layer:

This fundamental layer serves as an intermediary between apps and devices. It does several things, like:

(a) Device Management: It is a process of Finding, registering, and managing linked devices.

(b) Data acquisition: It is the process of getting data from devices and maybe pre-processing it to make it more efficient or secure.[9]

(c) Data processing: It is the process of preparing data for

use by applications by cleaning, filtering, aggregating, and possibly transforming it.

(d) Protocol translation: It is the process of converting data formats between various network protocols and devices.

(e) Security: Putting in place safeguards to protect sensitive data, such as access control and encryption.[10]

(f) Interoperability: The use of standard protocols to enable communication between devices made by various vendors.[7]

[4] Layer of Application:

The software programs in this layer make use of the information gathered from Internet of Things devices. Applications use Application Programming Interfaces, or APIs, to communicate with the middleware layer to access and control device data.[14]

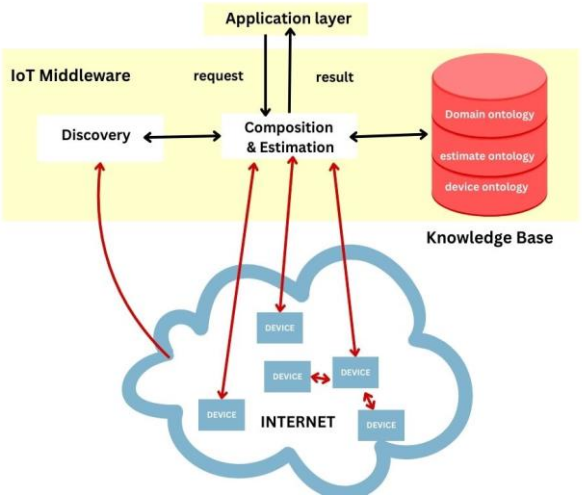

Fig. 3.1(a) Architecture of IoT Middleware

The above Cloud-based architecture of IoT Middleware can be explained as follows:

[1]Cloud database: This keeps and retrieves information from different Internet of Things devices.[15]

[2]Applications at the Application Layer: These are the programs that make use of the information gathered from Internet of Things devices. It uses request-response mechanisms to communicate with the middleware layer.[6]

[3] IoT Middleware layer:

(a) Discovery: The discovery functional block is in charge of locating and adding new devices to the Internet of Things.[3]

(b) Composition & Estimation: This part collects information from different devices and might merge it with information from other sources. Based on the information at hand, it may also estimate the missing data points.

(c) Knowledge Base: This part contains information and guidelines regarding the devices and the domains in which they are used.[12]

(d) Device: In an Internet of Things network, this is a digital representation of a physical device.[28]

3.2. Working of IoT Middleware

IoT middleware acts as an intermediary layer that bridges the gap between the sensor and actuator layer, the internet layer, and the application layer.[8] It essentially translates data between these layers and provides a set of services that streamline communication and data management within an IoT ecosystem.[37]

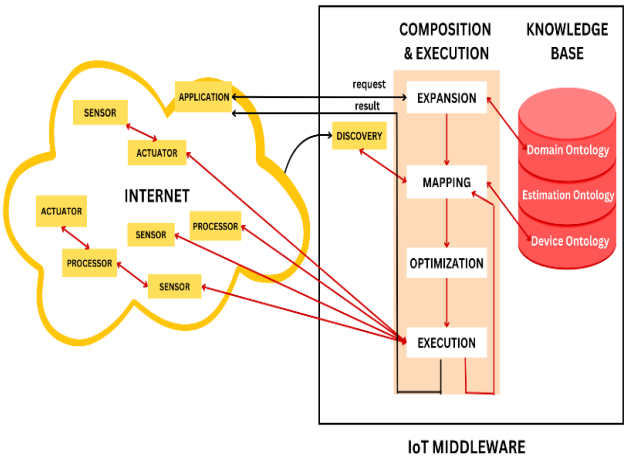

Fig. 3.2 (a) Working of IoT Middleware

Based on the labeled elements in the illustration, the following is a description of an IoT middleware's fundamental features:

[1] Device Ontology: It is a set of information that describes the characteristics and capabilities of different devices connected to the network. The middleware can identify and comprehend the different kinds of data that each device produces thanks to this ontology.[13]

[2]Sensors and Actuators: Physical devices that gather and transmit data or receive and carry out commands, respectively, are represented by sensors and actuators. Actuators can be thermostats, smart plugs, or valves, sensors can be motion detectors, temperature sensors, or traffic cameras.

[3] Discovery: The process of locating and registering devices on a network is known as discovery. In addition to keeping a directory of all connected devices, the middleware makes it easier for new devices to register.

[4] Mapping: The process of converting data between devices and apps is represented by mapping. Devices frequently use different protocols to communicate, and middleware helps ensure that data is formatted correctly for each recipient.

[5] Estimation Ontology: This is a body of information that

describes how sensor data should be processed and analyzed. The middleware can carry out operations like anomaly detection and removing unnecessary data thanks to this ontology.[13]

[6] Knowledge Base: A knowledge base can be used to analyze data streams, spot trends, and anticipate future events by storing past data and domain-specific knowledge.

[7] Execution: The middleware's ability to take actions in response to the data it receives is referred to as execution. For example, based on sensor data, an irrigation system may be set up to automatically activate sprinklers when soil moisture levels drop below a predetermined threshold.

[8] Composition & Estimation: The term "Composition & Estimation" describes how the middleware can compile information from various sensors and possibly merge it with knowledge base information to generate estimates or gain new insights.

[9] Request and Result: The communication between the middleware layer and the application layer is represented by the terms Request and Result. The middleware may be queried by an application to obtain historical data for analysis or real-time sensor data.

3.3. Basic Workflow of IoT Middleware for Critical Applications

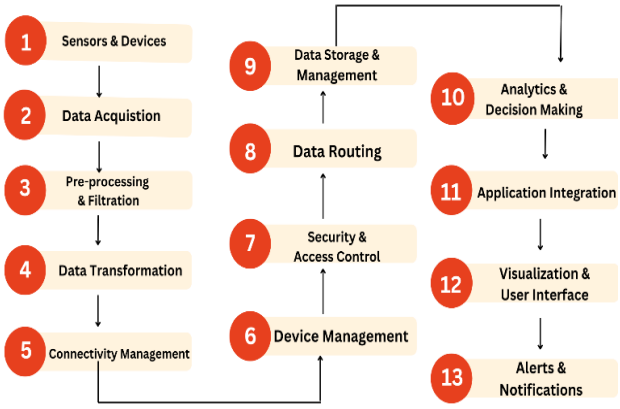

Fig. 3.3 (a) Workflow of IoT Middleware

An IoT middleware platform for critical applications works in the following steps, as illustrated:

[1] Sensors and Devices: Data is gathered and sent to the middleware platform by these physical components. These sensors are probably going to be high-precision devices that produce a lot of data in critical applications.

[2] Data Acquisition: Information is gathered from multiple sensors and devices by the middleware. Reliable data acquisition is necessary for critical applications to guarantee that decisions are based on correct and current information.[26]

[3] Pre-processing and Filtering: To reduce noise or prepare the data for additional analysis, the middleware may pre-process the data. To preserve data integrity in critical applications, data pre-processing may include removing outliers or accounting for sensor drift.

[4] Data Transformation: The middleware may change the data so that applications can use it in a certain format. This could entail combining data from several sensors or converting measurement units. Accurate data transformation is necessary in critical applications to prevent errors from entering the data stream.

[5] Connectivity Management: Durable and reliable connectivity between devices and the middleware platform is often required for critical Internet of Things applications. To keep devices online and data flowing uninterrupted, middleware constantly checks and maintains network connections.

[6] Device Management: The middleware makes it easier to communicate with connected devices by keeping track of them in a directory. Device management may include features such as provisioning for secure device registration and authentication and device health monitoring in critical applications.

[7] Security and Access Control: When it comes to important applications, security is crucial. To protect sensitive data from illegal access or cyberattacks, this step refers to the middleware's ability to implement encryption and access control mechanisms.[25]

[8] Data Routing: The middleware makes intelligent decisions about where to send the gathered sensor data. This routing may use redundant paths or prioritize important data streams in critical applications to guarantee delivery in the event of a network failure.

[9] Data Management and Storage: The collected data is safely stored by the middleware for subsequent analysis and retrieval. Important applications may have more stringent data governance guidelines and data retention policies.

[10] Analytics and Decision-Making: To process and analyze the gathered data in real-time, the middleware may interface with analytics tools. Advanced analytics may be used by critical applications to produce insights, spot trends, and aid in important decision-making.

[11] Application Integration: By enabling communication between different apps that make use of the gathered sensor data, the middleware helps. For secure and dependable information flow between various software systems in critical applications, this integration is necessary.

[12] User interface and visualization: The middleware may offer resources to help human operators see the sensor data. Real-time dashboards may be required by critical applications to promote situation awareness and allow operators to respond promptly.

[13] Notifications and Alerts: The middleware can send out notifications and alerts in response to predetermined criteria found in the sensor data. Timely notifications are crucial for critical applications as they alert operators to possible problems or equipment malfunctions.

4. Algorithms Used:

4.1. Algorithms for Making Efficient IoT Middleware in Critical Applications:

The following are some essential algorithms that help such middleware become efficient:

[1] Resource Optimization:

(a) Data compression algorithms: By compressing the sensor data before transmission, the processing load and bandwidth consumption can be reduced. Examples of such algorithms are LZMA and Huffman coding.

(b) Adaptive sampling: This algorithm modifies the rate at which sensor data are sampled in response to predetermined thresholds or current circumstances. By doing this, the volume of data is decreased for less important times while retaining all relevant details.

[2] Scalability:

(a) Algorithms for load balancing: Use algorithms such as the least connection or round-robin to split the load of data processing and communication among several middleware instances. This guarantees seamless operation during high data loads.

(b) Clustering algorithms: To group similar devices geographically or functionally, use clustering algorithms, such as k-means or DBSCAN. This lowers the total network traffic and enables localized processing.[39]

[3] Caching and Message Queuing:

(a) Less Recently Used (LRU): Use LRU caching techniques to give priority to data that is accessed frequently, which will decrease database access and speed up response times.

(b) Priority algorithms for queuing: Prioritize important messages in message queues using algorithms like priority queues to make sure they are handled first in the event of congestion.

4.2. Algorithms for Making Maintainable IoT Middleware in Critical Applications:

Several important algorithms that can help keep IoT middleware maintainable in crucial applications are as follows:

[1] Modular Design and Error Handling:

(a) Finite State Machines (FSMs): In the middleware, use FSMs to simulate device behavior and message flow. Specifying expected states and actions for unexpected ones enables clear state transitions and streamlines error handling.

(b) Exception handling algorithms: Put into practice clearly defined exception handling algorithms that classify errors, efficiently log them, and, depending on the kind of error, initiate the relevant recovery mechanisms.

[2] Configuration Management and Monitoring:

(a) Algorithms for self-configuration: To minimize human intervention and possible mistakes, use algorithms such as DNS-SD or Zeroconf (Bonjour) for automatic device discovery and configuration.

(b) Algorithms for anomaly detection: Use techniques such as machine learning-based outlier detection or statistical anomaly detection to find anomalous system behavior that may point to possible problems. As a result, proactive maintenance can be done before problems arise.

4.3. Algorithms for Making Reliable IoT Middleware in Critical Applications:

In crucial applications, the reliability of IoT middleware is dependent on algorithms that guarantee seamless functioning, accurate data, and prompt decision-making. The following is an overview of some important algorithms to achieve reliability:

[1] Error Handling and Recovery:

(a) BFT (Byzantine Fault Tolerance) algorithms: Algorithms like PBFT (Practical Byzantine Fault Tolerance), allow system failures in distributed environments. Even if some nodes malfunction, these algorithms guarantee consistent data and system state.

(b) Data replication algorithms: Put into practice algorithms to replicate data between several servers or devices. Due to redundancy, data availability is guaranteed even in the event of a node or device failure.

(c) Algorithms for forward error correction (FEC): Use FEC algorithms to add redundant data to transmitted data, such as Reed-Solomon coding. This ensures data integrity even in noisy communication channels by enabling error detection and correction.

[2] Security:

(a) Cryptographic algorithms: For both in-transit and at-rest data encryption, use strong cryptographic algorithms such as AES (Advanced Encryption Standard). If attackers manage to intercept sensitive data, this prevents unauthorized access.

(b) Key management algorithms: To create safe communication channels without pre-shared secrets, use secure key management algorithms like the Diffie-Hellman key exchange.

(c) Algorithms for intrusion detection: Use techniques such as machine learning-based intrusion detection or statistical anomaly detection to find malicious activity occurring within the network. This aids in preventing illegal access and security lapses.

[3] High Availability (HA):

(a) Algorithms for leader election: If a primary node fails in a cluster, use algorithms such as Paxos or Raft to elect a leader node. This guarantees uninterrupted operation even in the event of network outages or partitions.

(b) Heartbeat algorithms: Make use of heartbeat algorithms, in which nodes communicate their health to a central coordinator regularly. This makes failover mechanisms easier to implement and enables the prompt detection of node failures.

5. Methodology

Methodologies used to increase the efficiency, maintainability, and reliability of IoT Middleware in critical applications like healthcare, military, robotics, nuclear plants, etc. are listed below. It is crucial to ensure efficient, maintainable, and reliable IoT middleware is crucial in critical applications where even a small glitch can have disastrous effects. Here are a few approaches to help us do this:

5.1. Methodology for Efficient IoT Middleware:

[1] Optimize our resources: To reduce processing overhead on devices with limited resources, use lightweight protocols and data standards.

[2] Scalability: Build the middleware such that it can accommodate different numbers of devices and data volumes without experiencing a drop in performance. Methods such as microservices and containerization are useful.

[3] Message queuing and caching: Use message queues to manage varying data flows and caching techniques to save frequently accessed data.

5.2. Methodology for Maintainable IoT Middleware:

[1] Modular design: Divide the middleware into distinct, self-contained modules with understandable interfaces. This makes updates, future improvements, and troubleshooting simpler.

[2] Configuration management: Define system parameters using configuration files to facilitate updates without changing the code.

[3] Logging and monitoring: To rapidly detect and diagnose problems, put in place extensive logging and monitoring tools.

5.3. Methodology for Reliable IoT Middleware:

[1] Formal methods: Use formal verification techniques to demonstrate mathematically that crucial middleware functionalities are correct.

[2] Error control and recovery: Provide strong error control systems to adapt to unforeseen circumstances and quickly recover from malfunctions. It is essential to use strategies like failover mechanisms and message retries.

[3] Security: To safeguard data integrity and system access, put strong security measures like encryption, authentication, and authorization into place.

[4] High availability (HA): To guarantee continuous operation even in the event of hardware or software failures, design the middleware with redundancy in mind, incorporating features like load balancing and clustering.

6. Results

Based on the above Methodologies, we can conclude the most useful algorithms that can make IoT Middleware more Efficient, Reliable, and Maintainable for Critical Applications like Healthcare, Military, Agriculture, and Smart Cities, etc.

6.1. Algorithms For Highly Efficient IoT Middleware for Critical Applications:

[1] For Healthcare: Several algorithms are used in healthcare to increase efficiency. Critical patient data is managed by priority queues; frequently accessed records are accessed more quickly through caching; patient data patterns are identified by clustering; large datasets are compressed; tasks are distributed via load balancing; and resource utilization is optimized through adaptive sampling, which modifies data collection.[11]

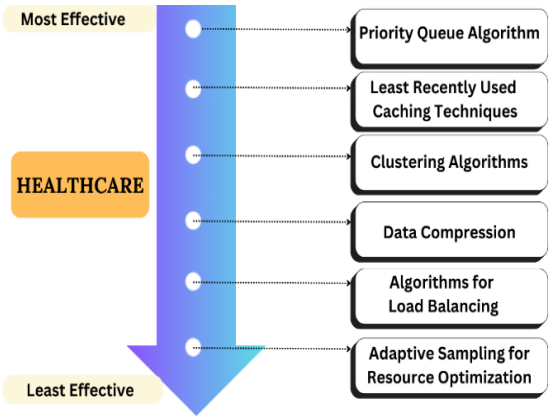

Fig. 6.1 (a) Ranking of Algorithms Used to Improve the Efficiency of IoT Middleware in Healthcare

(a) Priority queue algorithm: Vital for managing critical patient data and emergencies in real-time, ensuring urgent cases are addressed promptly.

(b) Least recently used caching technique: Useful for quick access to frequently accessed patient records or medical data, reducing latency in accessing critical

information.

(c) Clustering algorithms: Valuable for analyzing patient data to identify patterns or trends in diagnoses, treatment outcomes, or disease outbreaks.

(d) Data compression algorithms: Beneficial for transmitting medical imaging data or large datasets efficiently, especially in telemedicine applications or remote diagnostics.

(e) Algorithms for load balancing: Important for distributing computational tasks across healthcare infrastructure, ensuring efficient resource utilization in processing medical data.

(f) Adaptive sampling for resource optimization: Helpful for optimizing resource usage in dynamic healthcare environments, adjusting data collection rates based on patient conditions or workload variations.

[2] For Military: Peak efficiency in military operations is dependent on algorithms. Adaptive sampling optimizes sensor data collection based on situations, load balancing distributes tasks across networks, caching speeds up access to critical data, clustering aids in the analysis of reconnaissance data, compression reduces large files for transmission, and priority queues handle urgent information.[32]

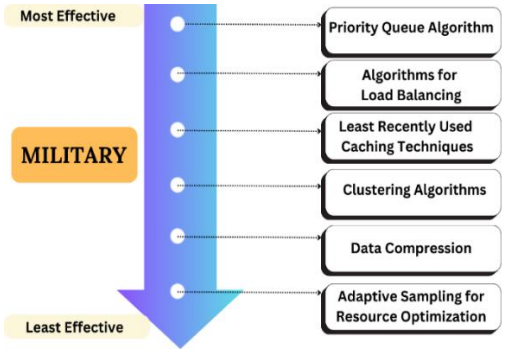

Fig 6.1 (b) Ranking of Algorithms Used to Improve the Efficiency of IoT Middleware in Military

(a) Priority queue algorithm: Military command and control systems must prioritize and process critical information in real-time, such as threat assessments or mission updates.

(b) Algorithms for load balancing: Essential for distributing tasks across military networks and resources efficiently, optimizing response times and resource utilization.[33]

(c) Least recently used caching technique: Valuable for quick access to frequently used intelligence data or mission-critical information, reducing latency in decision-making.

(d) Clustering algorithms: Useful for analyzing reconnaissance data or surveillance feeds to identify patterns, anomalies, or potential threats.

(e) Data compression algorithms: Important for transmitting large volumes of data, such as satellite imagery or sensor data, efficiently over constrained military networks.

(f) Adaptive sampling for resource optimization: Beneficial for optimizing sensor data collection in dynamic military environments, adjusting sampling rates based on mission requirements or threat levels.

[3] For Agriculture: Data-driven farm optimization relies on algorithms of differing degrees of efficiency. The quickest way to obtain frequently used data, such as the weather, is through caching. Adaptive sampling, which adjusts data collection based on urgent needs, comes after clustering, which aids in the analysis of patterns for well-informed decisions. Priority queues become crucial in emergency scenarios, while compression reduces the size of big data sets for transmission. The volume and complexity of the data in agriculture make load balancing less applicable in general.[29]

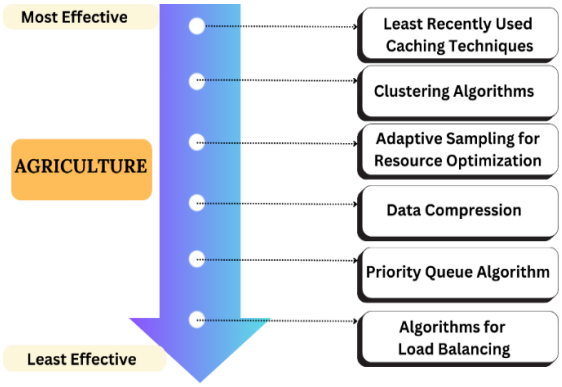

Fig.6.1 (c) Ranking of Algorithms Used to Improve the Efficiency of IoT Middleware in Agriculture

(a) Least recently used caching technique: Helpful for quick access to frequently accessed agricultural data, such as weather forecasts, soil moisture levels, or crop health information, reducing latency in decision-making.

(b) Clustering algorithms: Valuable for analyzing agricultural data to identify patterns in crop yield, disease outbreaks, or pest infestations, aiding in precision farming techniques.

(c) Adaptive sampling for resource optimization: Beneficial for optimizing irrigation schedules or fertilizer application rates based on real-time sensor data and environmental conditions.

(d) Data compression algorithms: Useful for transmitting agricultural sensor data efficiently over limited bandwidth networks, such as in remote farming locations.

(e) Priority queue algorithm: While less common in agriculture, it could be applied in scenarios where timely responses to crop emergencies or equipment failures are critical.

(f) Algorithms for load balancing: Typically, less relevant in agricultural contexts compared to other domains, as the scale and complexity of agricultural data processing may not require extensive load balancing techniques.

[4] For Smart Cities: The smooth operation of a smart city depends on several algorithms, each with varying degrees of efficiency. To ensure quick responses in emergencies, priority queues are essential. By distributing work across the city's infrastructure, load balancing maximizes resource utilization. Access to data that is frequently needed is sped up by cache.[35]

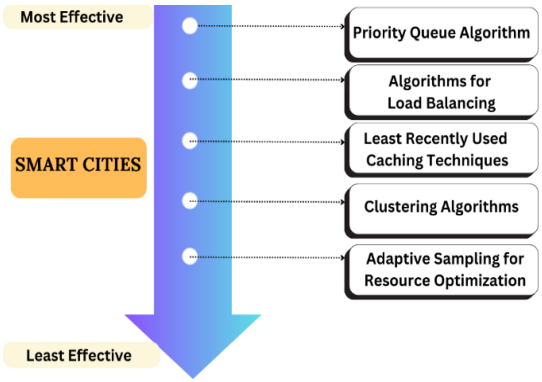

Fig. 6.1 (d) Ranking of Algorithms Used to Improve the Efficiency of IoT Middleware in Smart Cities

(a) Priority queue algorithm: Crucial for managing real-time events and emergencies in smart city systems, ensuring prompt responses to critical situations such as traffic accidents, public safety incidents, or infrastructure failures.

(b) Algorithms for load balancing: Essential for distributing urban services and managing resources efficiently across diverse infrastructure components in smart cities, optimizing traffic flow, energy distribution, and public services delivery.

(c) Least recently used caching technique: Valuable for quick access to frequently requested urban data, such as public transportation schedules, weather forecasts, or air quality information, reducing latency in urban service delivery and decision-making processes.

(d) Clustering algorithms: Useful for analyzing urban data to identify patterns, trends, or anomalies in areas such as traffic flow, energy consumption, waste management, or public health, facilitating data-driven urban planning and policy-making.

(e) Adaptive sampling for resource optimization: Beneficial for optimizing resource usage in dynamic urban environments, adjusting data collection rates based on real-time conditions, or changing demand patterns in areas like water management.

6.2. Algorithms For Highly Maintainable IoT Middleware for Critical Applications:

These algorithms when combined with the previously discussed methods (modular design, configuration management, logging, etc.) allow us to build highly maintainable IoT middleware for mission-critical applications.

[1] Layer of foundation with FSMs: Model device behavior and message flow with FSMs to create a strong base. This establishes an understandable and sustainable framework for the middleware's primary features.

[2] Improved maintainability through self-configuration: Automate device discovery and configuration by integrating self-configuration algorithms. In deployment and maintenance, this lowers the chance of errors and manual labor.

[3] Using anomaly detection algorithms to find unusual system behavior that might point to possible problems is a proactive maintenance strategy. This makes it possible to identify issues early on and address them before they worsen and become catastrophic failures.

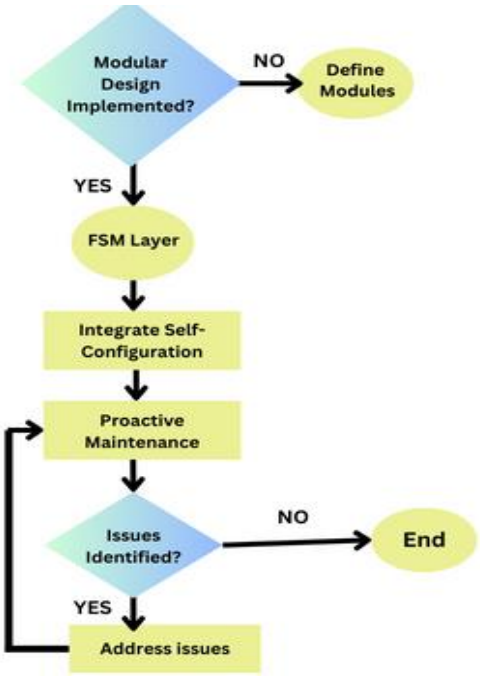

Fig. 6.2 (a) Use of Algorithms in Building Highly Maintainable IoT Middleware For Critical Applications.

6.3. Algorithms for Highly Reliable IoT Middleware for Critical Applications:

Especially in IoT middleware applications, Byzantine Fault Tolerance (BFT) algorithms are essential for ensuring the reliability and integrity of systems. In the Internet of Things situations where data security is crucial, BFT algorithms offer strong methods for identifying and resolving Byzantine faults, ultimately enhancing the system's overall reliability.

In the context of IoT middleware reliability, experimental validation of the Byzantine Fault Tolerance (BFT)

algorithm involves multiple crucial steps:

[1] Simulation Environment Setup: Creating a virtual environment that closely resembles the features of the intended IoT middleware program. This includes modeling network communication, introducing various fault scenarios, and building virtual nodes.[34]

[2]Algorithm Implementation: To guarantee the chosen BFT algorithm's fidelity to actual situations, it must be integrated into the simulated environment.[40]

[3] Fault Injection: Putting flaws into the system to assess how well the BFT algorithm performs in challenging circumstances. This includes emulating network splits, node failures, and Byzantine behaviors such as malicious nodes.

[4] Definition of Performance Metrics: Determining relevant performance metrics to assess the IoT middleware's dependability using the integrated BFT algorithm. Throughput, latency, consensus time, fault tolerance, and resilience to Byzantine faults are a few examples of possible metrics.

[5] Comparative Analysis: Running tests to compare how well the BFT algorithm performs with other approaches such as intrusion detection systems, cryptographic algorithms, or data replication algorithms. This makes it easier to evaluate the BFT algorithm objectively and determine how reliable it is for important IoT middleware applications.

[6] Scalability Assessment: Increasing the number of nodes and network topology complexity gradually to test the scalability of the BFT algorithm. This helps determine whether the algorithm can maintain reliability as the size and complexity of the system grow.

[7] Security Evaluation: Evaluating the BFT algorithm's resistance to various security risks, such as attacks meant to thwart consensus or jeopardize data integrity. Examining the algorithm's cryptographic characteristics and resilience to Byzantine behaviors is necessary to achieve this.

The above methodology outlines ways to improve efficiency, maintainability, and reliability in IoT middleware for critical applications. However, it does not directly compare a proposed system to an existing one. That said, we can analyze how the methodologies address common shortcomings of existing systems:

Existing System Shortcomings:

(a) Inefficient resource usage by devices with limited processing power.

(b) Inability to handle large numbers of devices and data volume without performance drops.

(c) Difficulty in updates, troubleshooting, and future improvements due to monolithic design.

(d) Prone to errors and slow recovery times.

(e) Security vulnerabilities.

How Methodology Addresses Shortcomings:

[1] Efficiency: Lightweight protocols, data standards, message queuing, and caching all contribute to reducing resource usage and handling data flow efficiently.

[2] Maintainability: Modular design with clear interfaces simplifies updates, troubleshooting, and future improvements. Configuration management through files enables updates without code changes. Logging and monitoring aid in problem detection and diagnosis.

[3] Reliability: Formal methods ensure core functionalities work as intended. Error control and recovery mechanisms like failover and retries help with quick recovery from malfunctions. Security measures like encryption protect data and system access.

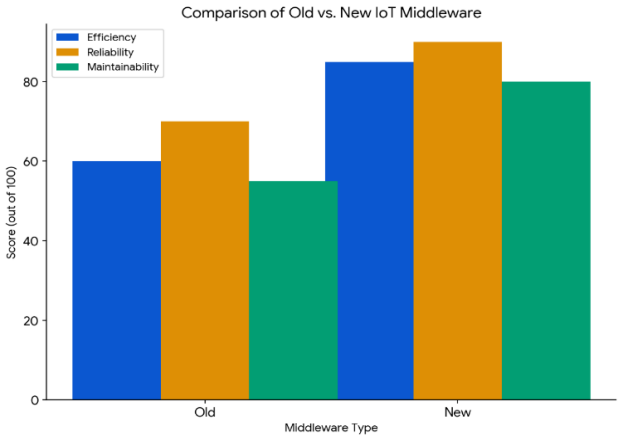

Fig 6.3 (a) Comparison Graph of Old IoT Middleware Vs New IoT Middleware

7. Future Scope

The study proposed above highlights the need for advancements in IoT middleware to address the challenges within the IoT ecosystem.

[1] Enhanced Real-time Processing and Scalability:

(a) Distributed and In-Memory Processing: For quicker real-time analytics, data will be processed in-memory or closer to its source (fog computing).

(b) Machine Learning at the Edge: To save bandwidth and provide quicker insights, devices will preprocess data at the edge using machine learning models.

[2] Robust Security Measures:

(a) Post-quantum cryptography: Post-quantum cryptography will be incorporated into middleware to guarantee future-proof security in light of the possible threat posed by quantum computers cracking existing encryption.

(b) Blockchain for Secure Data Sharing: In key applications, blockchain technology will allow for transparent and safe data sharing between devices and stakeholders. [21]

[3] Improved Resource Management:

(a) Ultra-low Power Communication Protocols: In important applications, protocols with low power consumption will be essential for battery-powered devices.

(b) Resource-aware Algorithms: Based on the resources that are available on devices, middleware will use algorithms to dynamically modify data collecting and processing.

[4] Universal Interoperability and Standardization:

(a) Standardised Data Formats and APIs: To facilitate smooth communication between devices and middleware from various vendors, open-source and interoperable data formats and APIs will eventually become the norm.

(b) Semantic Interoperability: This allows for improved data interchange and cross-application collaboration by enabling middleware to comprehend the meaning of data in addition to its format. [39]

8. Conclusion

The research leads to the conclusion that the scalability, security, and interoperability shortcomings of current IoT middleware solutions make them unable to satisfy the demands of critical applications. This research suggests approaches to create better middleware for vital applications in healthcare, the military, agriculture, and smart cities by concentrating on algorithms for efficiency, maintainability, and reliability. The suggested strategy provides a road map for developing reliable middleware for crucial applications by emphasizing efficiency (e.g., priority queues, caching), maintainability (e.g., FSMs), and dependability (e.g., BFT algorithms).

Subsequent developments in domains such as instantaneous processing, flexible security, and complex context-aware decision assistance will additionally enable crucial applications to function with enhanced independence and effectiveness. This will result in significant improvements across multiple industries that depend on the Internet of Things.

References

[1] Adib Mehedi, Adib Hassan Tokee, Surovi Islam, Md Saef Ullah Miah (2020), IoT Based Healthcare Middleware, ICCA: Proceedings of the International Conference on Computing Advancements, 2020.

[2] Omar Hernando Moreno Torres; Javier Antonio Ballesteros Ricaurte (2015), Application Middleware for Management of Medical Applications Based on HL7 Standards. Asia-Pacific Conference on Computer Aided System Engineering.

[3] Sunitha Lingam (2021), IoT Healthcare Applications. The Fusion of Internet of Things, Artificial Intelligence, and Cloud Computing in Health Care pp 135–154

[4] Ghofrane Feresi (2020), Study of Middleware for Internet of Healthcare Things and Their Applications. International Conference on Smart Homes and Health Telematics ICOST The Impact of Digital Technologies on Public Health in Developed and Developing Countries pp 223–23

[5] Venki Balasubramanian, Alireza Jolfaei (2021), A scalable framework for healthcare monitoring application using the Internet of Medical Things. Software - Practice and Experience (John Wiley & Sons, Ltd)-Vol. 5

[6] Rita Zgheib, Emmanuel Conchon, Rémi Bastide (2019), Semantic Middleware Architectures for IoT Healthcare Applications. Part of the Lecture Notes in Computer Science book series (LNISA, volume 11369)

[7] Madhavi Latha Challa, K. L. S. Soujanya, C. D. Amulya (2020), Remote Monitoring and Maintenance of Patients via IoT Healthcare Security and Interoperability Approach. , Cybernetics, Cognition and Machine Learning Applications pp 235–245

[8] José Cecílio, Pedro Furtado (2014), Middleware Solution for HealthCare IoT Applications. International Wireless Internet Conference WICON 2014: Wireless Internet.

[9] R. Venkateswara Reddy, D. Murali, J. Rajeshwar (2019), Context-Aware Middleware Architecture for IoT-Based Smart Healthcare Applications. Innovations in Computer Science and Engineering

[10] Pedro Maia, Augusto Baffa, Everton Cavalcante, Flavia C. Delicato, Thais Batista, Paulo F. Pires (2015), A Middleware Platform for Integrating Devices and Developing Applications in E-Health. Brazilian Symposium on Computer Networks and Distributed Systems (SBRC)

[11] Manisha Banka, Ankita Mishra, Sushreeta Tripathy (2022), IoT in Healthcare Domain. Graduate Research in Engineering and Technology (GRET)

[12] Siddhant Mukherjee, Dayaram Sonawane & Kalyani Bhole (2018), Design and Development of Scalable IoT Framework for Healthcare Application.

International Conference on ISMAC in Computational Vision and Bio-Engineering

[13] Abdullah Alamri (2018), Ontology Middleware for Integration of IoT Healthcare Information Systems in EHR Systems. MDPI.

[14] T. Jayasri, Kurunji Malar. R, Aruna Singaravelu (2015), WSN Compatible IOT middleware for healthcare monitoring system. Research, 2015

[15] Istabraq M. Al-Joboury, Emad H. Al-Hemiary (2018), Internet of Things Architecture Based Cloud for Healthcare. IJICT.

[16] M. T. Moghaddam, É. Rutten & G. Giraud (2020), Protocol for a Systematic Literature review on Adaptive Middleware support for IoT and CPS, INRIA, 2020,HAL-02948347.

[17] Cédric Melançon, Guillaume Simard, Maarouf Saad, Kuljeet Kaur, Julien Gascon-Samson (2023), BlazeFlow: A Multilayer Communication Middleware for Real-time Distributed IoT Applications. Mid4CC '23: Proceedings of the 1st International Workshop on Middleware for the Computing Continuum, Pages 30-35

[18] M. Bazzani Davide Conzon (2021), Enabling the IoT Paradigm in e-health solutions through virus Middleware. Computer Science, Engineering, Environmental Science, Medicine 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications.

[19] R.A.K. Lakpriya, W.A.S. Rathsara, Piumi Fernando, H.S. Thenuwara, A. Senarathne (2022), Secure IoT Middleware using SDN & Intent-based Routing. International Conference for Convergence of Technology (I2CT).

[20] G. Bhandari, Andreas Lyth, Andrii Shalaginov, Tor-Morten Grønli (2023), Distributed Deep Neutral-Network Based Middleware for Cyber-attacks Detection in smart IoT ecosystem: A Novel Framework & Performance evaluation approach. Electronics 2023, 12(2), 298; https://doi.org/10.3390/electronics12020298

[21] Eryk Schiller, Elfat Esati, B. Stiller (2021), IoT Bases Access Management Supported by AI & Blockchains. Conference on Network and Service Management. DOI:10.23919/CNSM52442. 2021.9615523

[22] R. Tiburski, Leonardo A. Amaral, Everton de Matos, Dario F. G. de Azevedo, (2017), Evaluating the use of TLS & DTLS Protocols in IoT Middleware systems applied to e-health. Consumer Communications and Networking Conference

[23] Qamar Alfalouji, Thomas Schranz, A. Kümpel, M. Schraven, T. Storek, A. Monti, D. Müller, G. Schweiger (2022), IoT Middleware Platform for Smart Energy systems: An Empirical Expert. MDPI

[24] Shalmoly Mondal, P. Jayaraman, P. D. Haghighi & Dimitrios Georgakopoulos (2021), Modelling IoT Application Requirements for Benchmarking IoT Middleware Platforms. International Conference on Information Integration and Web-based Applications & Services.

[25] Guru Prasad Bhandari, Andreas Lyth, Andrii Shalaginov, Tor-Morten Groenli (2022), Artificial Intelligence enabled Middleware for Distributed Cyber-attacks Detection in IoT- Based Smart Environments. IEEE International Conference on Big Data.

[26] Durga Amarnath M. Budida, Ram S. Mangrulkar (2017), Design & Implementation of Smart Healthcare System using IoT. International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)

[27] Amrita Rai, Deepti Sharma, Shubhyansh Rai, Amandeep Singh, Krishna Kant Singh (2021), IoT-Aided robotics development and applications with AI. Emergence of Cyber Physical System and IoT in Smart Automation and Robotics

[28] Bidyut Mukherjee, Roshan Lal Neupane, Prasad Calyam (2017), End-to-end IoT Security Middleware for cloud-fog communication. IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)

[29] Lal Verda Çakır, Tuğçe Bilen, Mehmet Özdem, Berk Canberk (2023), Digital Twin Middleware For smart farm IoT Networks. Communications and Networking (BalkanCom), International Balkan Conference.

[30] A. Steven, Suharnawi, F. Alzami, Filmada Ocky Saputra, Rama Aria Megantara & P. Andono (2023), Architecture Development of Middleware & API for IoT-based red onion farming. 2023 International Seminar on Application for Technology of Information and Communication (iSemantic)

[31] F. Li, L. Chen, Y. Wang, and X. Wang. (2022), Research on Military Application of Operating System for Internet of Things. Part of the Communications in Computer and Information Science book series (CCIS, volume 1509)

[32] Rune Langleite, Carsten Griwodz, and Frank T. Johnsen, Norwegian Defence Research Establishment (FFI), Kjeller, Norway (2021), Military Applications of Internet of Things: Operational Concerns Explored in Context of a Prototype Wearable. , International

Command and Control Research and Technology Symposium (ICCRTS)

[33] M. Tortonesi, A. Morelli, M. Govoni, J. Michaelis, N. Suri, C. Stefanelli, S. Russell, 1 Department of Engineering, University of Ferrara, Ferrara, Italy (2016), Leveraging Internet of Things Within the Military Network Environment - Challenges and Solutions. IEEE 3rd World Forum on Internet of Things (WF-IoT)

[34] Dey, E., Walczak, M., Anwar, M. S., & Roy, N. (2022), A Dependable and Low Latency Synchronizing Middleware for Co-simulation of a Heterogeneous Multi-Robot Systems. 32nd International Conference on Computer Communications and Networks (ICCCN)

[35] Vikas K. Kolekar, Yashwant Wankhade (2023), Smart City IoT Data Management with Proactive Middleware.

[36] Poorna Chandra Tejasvi, Vasanth Rajaraman, Arun Babu Puthuparambil, & Bharadwaj Amrutur (2020), Vermillion: A High-Performance Scalable IoT Middleware for Smart Cities.

[37] Amirhossein Farahzadi, Pooyan Shams, Javad Rezazadeh, Javad Rezazadeh, & Reza Farahbakhsh (2017), Middleware technologies for the cloud of things: a survey.

[38] Charith Perera, Arkady Zaslavsky, Peter Christen, & Dimitrios Georgakopoulos (2014), Context-Aware Computing for The Internet of Things: A Survey.

[39] Y. Dongre and P. D. Patil, "An Analysis of Heterogeneous Device Middleware for Quality Metrics," 2023 International Conference on Emerging Smart Computing and Informatics (ESCI), Pune, India, 2023, pp. 1-6

[40] Dongre, Y. ., & Patil, P. . (2023). Genetic Algorithm based Optimal Service Selection of Composition in Middleware using QoS Correlation. International Journal of Intelligent Systems and Applications in Engineering, 11(2), 20–29.