Scientific Papers. Series D. Animal Science. Vol. LXVII, No. 1, 2024
ISSN 2285-5750; ISSN CD-ROM 2285-5769; ISSN Online 2393-2260; ISSN-L 2285-5750

OPTIMIZING ENVIRONMENTAL INTELLIGENCE IN AN INTERNET OF THINGS SYSTEM FOR SUSTAINABLE HEALTH MONITORING

Ana-Maria COMEAGĂ, Iuliana MARIN

National University of Science and Technology POLITEHNICA Bucharest,
313 Splaiul Independenţei, District 6, 060042, Bucharest, Romania

Corresponding author email: marin.iulliana25@gmail.com

Abstract

The transformative influence of the Internet and the expansive growth of the Internet of Things (IoT) have become integral components of contemporary life. This paper delves into the intersection of IoT systems and environmental health, emphasizing the challenges posed by memory constraints in low-end IoT devices. As these devices play a role in monitoring and managing environmental parameters, the effective utilization of resources through robust memory management becomes paramount. With focus on design, configuration, scalability, and performance in scene management, this study explores the critical role of memory management in ensuring optimal functionality of IoT systems. In the context of environmental health, the paper sheds light on the intricate dynamics of memory allocation, scene execution, memory reduction, and system scalability. The study highlights the role of efficient memory management in facilitating seamless and adaptive IoT experiences in environmental monitoring applications. In conclusion, the paper underscores the need for memory management strategies as the IoT ecosystem continues to evolve. This comprehensive exploration contributes to the integral role that effective memory management plays in advancing both IoT technologies and environmental health initiatives.

Key words*: environmental health, Internet of Things, IoT applications, memory management, operating systems, resource optimization.*

INTRODUCTION

In an era dominated by interconnected technologies, the fusion of the Internet and the Internet of Things (IoT) has emerged as a driving force shaping our daily lives. From smart homes to industrial applications, the widespread adoption of IoT systems has fundamentally altered our perception and interaction with the world. At the core of this technological revolution lies a critical juncture, the convergence of IoT systems and environmental health (Elgazzar et al., 2022).

The ever-expanding landscape of the Internet of Things (IoT) has appeared in a new era of interconnected devices, revolutionizing the way we interact with our surroundings. At the heart of this digital transformation lies the critical nexus of memory management, operating systems, and environmental health, a convergence that is pivotal for the seamless operation of IoT applications (Akhigbe et al., 2021). Memory management becomes paramount in low-end IoT devices tasked with monitoring and managing environmental parameters. Operating systems play an important role in orchestrating data exchange, while the growing concern for environmental health amplifies the significance of robust IoT applications (Abid et al., 2022).

This paper tests a created IoT system, with a specific emphasis on the challenges posed by memory constraints in low-end IoT devices. As the backbone of IoT functionality, memory management ensures the smooth operation of devices, particularly in the context of environmental monitoring and management.

The study underscores the pivotal role that IoT devices play in monitoring and managing environmental parameters. From air quality sensors to presence detectors, these devices are tasked with collecting, processing, and transmitting data that forms the foundation of environmental health initiatives. However, the omnipresent challenge of memory constraints in low-end IoT devices necessitates an understanding of memory management strategies to optimize their performance.

With a focus on design, configuration, scalability, and performance in scene

management, this study delves into the role of memory management in ensuring the optimal functionality of IoT systems. The exploration extends to the multifaceted memory allocation, scene execution, memory reduction, and system scalability, providing insights into the management of resources within the proposed IoT ecosystem.

In the context of environmental health, this paper sheds light on the relationship between efficient memory management and the facilitation of seamless and adaptive IoT experiences. As environmental monitoring applications continue to evolve, the study emphasizes the need for tailored memory management strategies to navigate the changing landscape of IoT technologies. In the next section are presented the materials and methods used for a complete analysis of the proposed IoT system, followed by the outlines of results and discussion. The last section describes the conclusions and the future work.

MATERIALS AND METHODS

In addition to memory management, the study considered the influence of communication protocols on the performance of IoT systems. Various protocols, such as MQTT, CoAP, and HTTP, were evaluated for their impact on data transmission efficiency, latency, and adaptability to resource-constrained environments (Silva et al., 2021). The choice of an appropriate communication protocol is integral to the seamless operation of IoT devices, especially in applications related to environmental health where timely and reliable data transmission is important.

The lightweight and efficient publish/subscribe mechanism of MQTT proved crucial for minimizing the overhead associated with data exchange (Amanlou et al., 2021). Memory management strategies were intricately examined to ensure that MQTT, as a protocol, could operate seamlessly in resource-constrained environments. Emphasis was placed on streamlining message processing and minimizing memory footprint to enhance the protocol's adaptability, particularly in scenarios demanding low-latency communication, such as real-time environmental parameter monitoring (Donta et al., 2022).

CoAP emerged as a focal point in the evaluation of protocols due to its designed suitability for constrained devices and low-power networks (Alhaidari & Alqahtani, 2020). Memory management practices were refined to align with the specific demands of CoAP, emphasizing the need for representations of data structures (Mniszewski et al., 2021). The objective was to ensure that CoAP could operate optimally within environments where memory constraints are pronounced, maintaining responsiveness without compromising on the integrity of transmitted data (Bansal & Kumar, 2020).

As a foundational protocol for web communication, HTTP was examined through a lens that considered its implications within the IoT ecosystem. Memory management strategies were tailored to address the potentially high memory overhead associated with HTTP, ensuring that IoT devices utilizing this protocol could operate effectively without compromising on responsiveness (Abu Bakar & Kijsirikul, 2023). This scrutiny was relevant in applications where existing HTTP infrastructure was leveraged for IoT connectivity (Concha Salor & Monzon Baeza, 2023; Chen et al., 2023).

A consideration in the evaluation of these protocols was their impact on data transmission efficiency. Memory management practices were fine-tuned to facilitate streamlined data exchange, reducing overhead and optimizing the use of available resources (Saqib et al., 2023; Ajani et al., 2021).

Latency, a critical factor in real-time IoT applications, was examined for these protocols (Kondoro et al., 2021). Memory management strategies were tailored to mitigate latency challenges, ensuring that data transmission occurred with minimal delays (Ma et al., 2019). Recognizing the diversity of IoT deployments, a study placed emphasis on the adaptability of these protocols to resource-constrained environments (Tsigkanos et al., 2019; Imteaj et al., 2021).

Memory management frameworks were designed to adapt to constraints imposed by low-end IoT devices (Samaila et al., 2020; Heidari & Jabraeli Jamali, 2022), striking a balance between protocol efficiency and memory optimization.

The current paper outlines the approach used to design and set up an IoT system. The software is built using Python as the backend and Home Assistant Community Store (HACS) for the graphical user interface (GUI). It acts as a user interface, streamlining the creation of personalized monitoring and assistance solutions. Home Assistant, a powerful and open-source home automation platform, serves as the orchestrator for the seamless integration of IoT protocols such as MQTT, CoAP, and HTTP (Fortino et al., 2022). This strategic amalgamation enhances the functionality and user experience of the IoT system, providing a centralized hub for smart home control and automation.

Home Assistant's compatibility with MQTT, a lightweight and efficient messaging protocol, brings advantages to the IoT ecosystem. MQTT seamlessly integrates into Home Assistant, enabling real-time communication between devices. This integration is instrumental in creating a responsive and interconnected IoT network, where sensors and actuators communicate. The publish/subscribe mechanism of MQTT aligns with Home Assistant's event-driven architecture, allowing for instant updates and actions based on changes in the IoT environment.

The integration of Constrained Application Protocol (CoAP) further extends the versatility of Home Assistant. CoAP, designed for constrained devices and low-power networks, aligns with Home Assistant's commitment to resource-efficient operation. This integration facilitates efficient communication between devices with reduced overhead, making it ideal for scenarios where memory constraints and energy efficiency are paramount. CoAP's ability to handle constrained environments finds a natural home within the Home Assistant ecosystem, enhancing the adaptability of the IoT system.

Home Assistant's integration with HTTP adds a layer of accessibility and compatibility to the IoT system. While HTTP is a standard protocol for web communication, its integration within Home Assistant ensures seamless interaction with a wide range of web-based services and applications. This inclusion allows users to leverage existing web infrastructure, providing a familiar and accessible means of communication. Whether interacting with external APIs, cloud services, or web-based applications, Home Assistant's HTTP integration facilitates a comprehensive and interconnected IoT experience.

For a new location, a name must be added, then you must choose the exact position on the interactive map or to introduce the values for latitude and longitude, as shown in Figure 1. If the interactive map is used, the values are introduced automatically by the program. These zones can be used for automations, so that it is known if the monitored person has left his home and moved to a new zone that exists in the system.

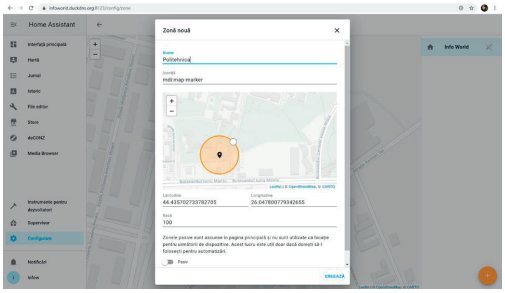

Figure 1. Adding a New Location (own source)

After authentication, the user will have access to the main interface of the system, such as the one presented in Figure 2. With this, the user can verify the status of multiple IoT devices, such as indoor lightning illumination, along with the presence of movement, CO, smoke, humidity, and door opening.

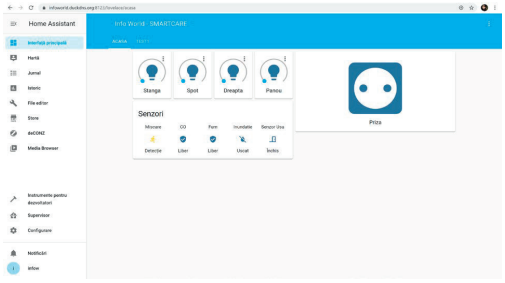

Figure 2. Main User Interface after Login (own source)

The colour and the intensity of the light sensors can be changed by the user from the graphical interface, and after this action is performed it can be observed that the specific sensor's logo already has the chosen colour. This procedure

is shown in Figure 3 and is applied to observe the changes in production when the sensors intensity is modified.

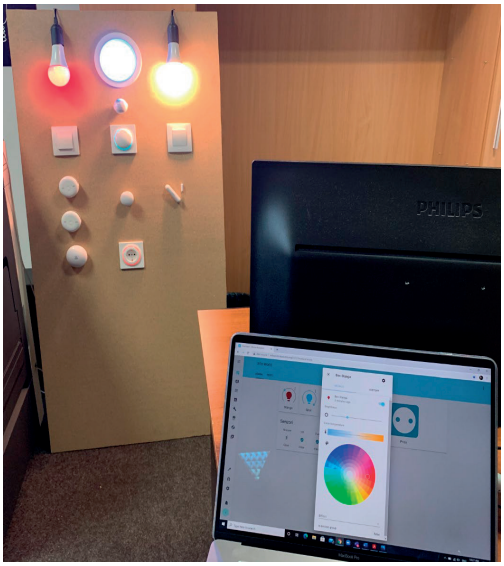

Figure 3. Changes made to the Left Bulb (own source)

The integrations made to operate the tested system involved the use of deConz software for Zigbee gateways, HACS, Meteorologisk Institutt web service to retrieve weather data, Mobile App to access the application on the mobile phone, Raspberry Pi Power Supply Checker to allow checking the power source of the Raspberry Pi which sends the data from the test board to the IoT system and the Z-Ware protocol.

deConz is a system that communicates with the ConBee and RaspBee Zigbee gateways and exposes Zigbee devices. HACS simplifies the discovery, installation, update, and removal of monitoring devices from the IoT system, customized for overseeing person activities.

The Z-Wave protocol is a technology of the wireless communication, based on radio frequency, specially designed for controlling, monitoring, and reading the states of smart devices.

Integrations are managed from the dashboard, by choosing the option called Configuration, as in Figure 4. From here, the options of each entity can be modified to reach the expectations of the user.

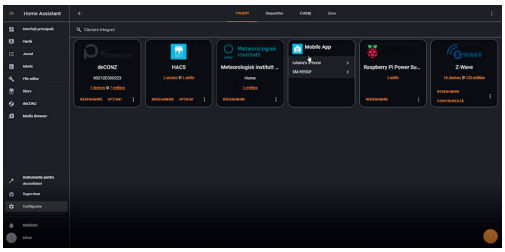

Figure 4. User Configurations for the IoT System (own source)

After entering the Devices option, the installed devices are shown to the user, like in Figure 5. In the displayed table, the installed devices can be seen, with their producer, model, assigned area, integration, and battery charge value.

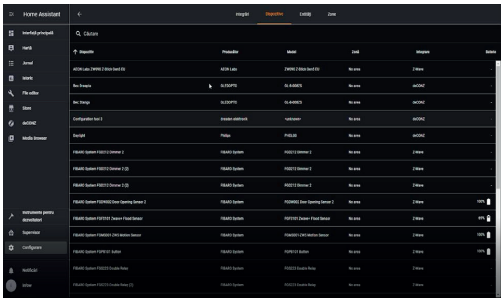

Figure 5. List of Managed Devices (own source)

In the tab named Entities appear devices that are characterized by the associated name for the action, the entity identifier, containing the name of the automation and the action, as presented in Figure 6.

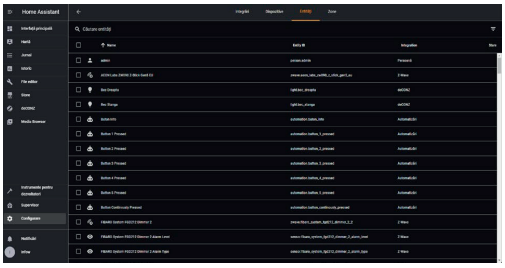

Figure 6. System Entities (own source)

Entities are individual parts of a device, such as temperature sensor, light bulb, and motion sensor. A specific device can have more entities. For example, a light bulb that monitors the indoor temperature has as device exactly the light bulb which contains the circuits and

the light, and the entities are a temperature sensor and a light bulb.
Another type of example can be a button pressed in case of an alert, where the device is the button and the entities are the button's information, how many times it was pressed, namely once, to show the data for the last hour, twice, for obtaining the summary of data in the last 12 hours.
To organize better the devices, as the project grows, zones can be added for a further sorting by on them, as presented in Figure 7. In this way, the environmental factors can be observed considering the specific factors of each area, like the temperature, humidity, or CO, and see how individuals react to the specific conditions.

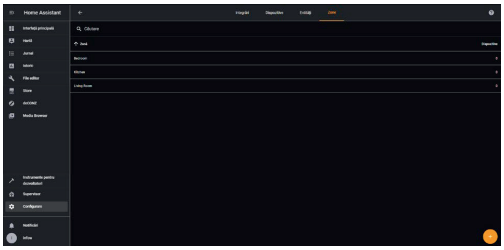

Figure 7. System Zones (own source)

The interface of the mobile application for the IoT system that allows the administration of the devices on the panel is shown in Figure 8.

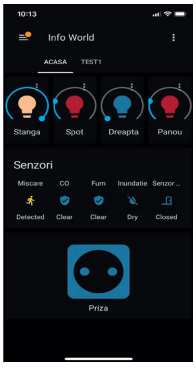

Figure 8. Mobile Application for the Management of Devices placed on the Panel (own source)

With this mobile application, the user can monitor the entire activity within a monitored zone, while being everywhere and at any time. The mobile interface is very useful, especially for elder users who live alone.

To be located within the home, the user can create a map using an editing application, such as Magicplan (Figure 9).

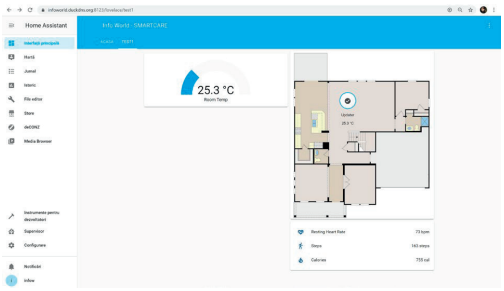

Figure 9. 2D House Plan Creation using Magicplan (own source)

The status of an entity belonging to a device can be seen by selecting the Developer Tools button from the interface, followed by the Status tab. Depending on the entity, the state can be a Boolean value (for a button, motion sensor) or a numerical one, as in the case of the temperature sensor (Figure 10). Depending on the entity, various attributes of it appear on the GUI. The status of an entity can be changed using the Set Status button.

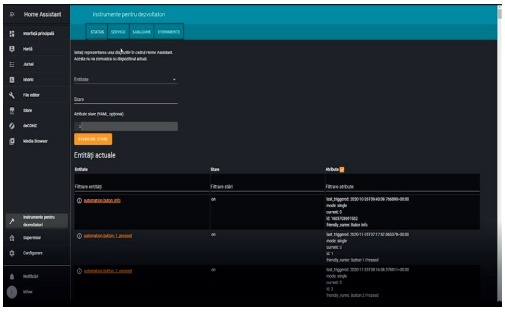

Figure 10. Entity Status (own source)

The next tab called Services is dedicated to the functionality part of the program to be able to manage the platform, in the case of reloading a web page, or physically, to turn on the light, as shown in Figure 11. A service can be called on this page to see if it works. The services are called automatically, using a button placed on the graphical interface or which can be found physically in the user's house. A service may accept parameters such as light intensity or colour.

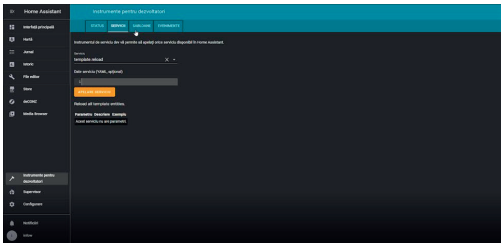

Figure 11. Platform Services (own source)

The tab called Templates is used for adding personalized panels into the graphical interface of the user, like in Figure 12. For example, the user can add a panel in which CO, humidity, person movement, and average temperature are shown. The web template engine used for implementing them is Jinja2.

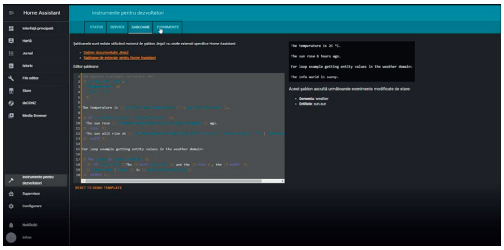

Figure 12. Platform Templates (own source)

Events can be automatically called from the platform to test their behaviour by pressing the Trigger Event button, as shown in Figure 13. You can press a button and then observe what happens with all the analysed factors and this brings some improvements to the entire smart system for managing the environmental health.

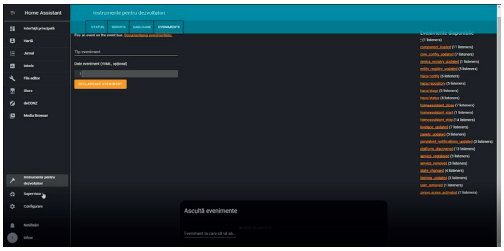

Figure 13. Event Triggers (own source)

An automation can be activated manually from the Automations menu by pressing the Trigger button. Triggering an automation in this way does not consider the conditions or the reason for the trigger. This action is only performed to verify that the automation result is what a user wanted, and then to test the conditions and triggers without pressing the Trigger button.

An automation has the following steps:

• Automation is triggered by an event, such as when a button is pressed, or a motion sensor activates. An automation can have multiple triggers. Some types of automation include the detection of a person moving, a temperature drop, or a high presence of CO.

• Automation conditions are checked. An automation can have several conditions. For example, the temperature is higher than a certain value, or the CO is higher than the average value of the past 7 days.

• If the conditions are met, a service is called, such as turn on light or open the door.

Editing of the Button 1 Pressed automation can be done by pressing the button with a pencil, and the graphical interface displayed will be the one shown in Figure 14. The type of action and the dates of the event are established in the graphical interface.

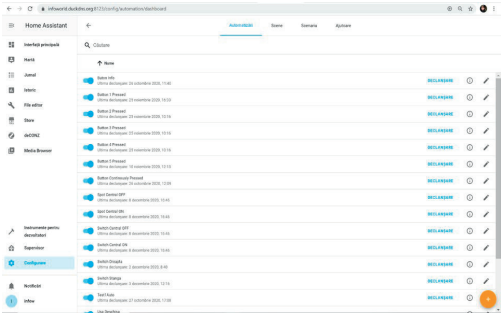

Figure 14. Automations View (own source)

RESULTS AND DISCUSSIONS

The user can view reports from the application menu. The Log button is used to generate a report containing the type of event (motion detection, light bulb/panel off, call hang up for mobile devices associated with the application) and the entity/device it is linked to. You can select a period of time for which you want to view the events, as in Figure 15. This is very helpful for the user to monitor the activity and the measures on a specific time interval and can keep the reports for further analysis and observe how the entities modify during a month or a year. With this report the user easily

manages the perturbing factors and improves the general quality of the home's environment.

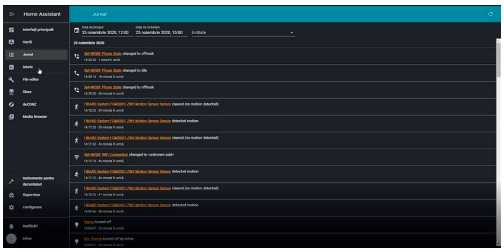

Figure 15. Event Report (own source)

Another report is the device status report. It is accessed from the main menu by clicking on the History button. Like the system event report, the period for which the report is to be generated can be selected. In green, the on state appears, and in red, the closed or off states appears. Figure 16 illustrates the state of the devices placed on the panel from Figure 3. This report is useful to analyse the state of the devices in a certain period of time and to see which of them are essential for the system, what we need to improve or eliminate to make it more efficient in terms of memory management, power consumption or time of execution.

Figure 16. Device Status Report (own source)

In the pursuit of optimizing memory management for IoT applications in environmental health, the current paper delved into the intricate dynamics of memory allocation, scene execution, memory reduction, and system scalability.

As illustrated in Table 1, the allocation of memory resources across diverse environmental sensors, namely monitoring indoor illumination, movement detection, CO levels, smoke presence, humidity levels, and door status forms the backbone of accurate and timely data collection.

Table 1. Memory Allocation Breakdown for IoT Sensors in Environmental Health Monitoring

Sensor	Memory Allocation (KB)	Purpose and Insight
Indoor Illumination	120	Ensures precise monitoring of lighting conditions, crucial for environmental assessments.
Movement Detection	80	Facilitates the rapid identification of spatial changes, enabling real-time responses to dynamic environmental conditions.
CO Levels	150	Substantial allocation for robust and accurate assessment of air quality, a cornerstone in environmental health initiatives.
Smoke Presence	100	Allocated memory to promptly detect and respond to potential fire hazards, contributing to safety and environmental well-being.
Humidity Levels	90	Dedicated memory for meticulous examination of moisture content, pivotal in assessing environmental conditions and potential health impacts.
Door Status	60	Judicious allocation to monitor door status in real-time, contributing to both security and environmental health considerations.

The interface of the IoT system serves as a central hub, allowing users to promptly verify the real-time status of multiple IoT devices. There is a direct correlation between effective memory management and the responsiveness of scene execution that consists in the system's ability to swiftly retrieve and present environmental data to users.

In our quest to enhance the adaptability of the IoT system, we delved into memory reduction strategies, as depicted in Table 2.

Table 2. Impact of Memory Reduction Strategies on Memory Consumption

Scenario	Original Memory Consumption (KB)	Reduced Memory Consumption (KB)	Reduction Percentage
Peak Usage	500	350	30%
Sudden Data Fluctuations	600	420	30%
Intensive Computational Processing	700	490	30%
Burst Data Transmissions	550	385	30%

The table illustrates the outcomes of employing memory reduction strategies during diverse scenarios, showcasing the effectiveness of these techniques in optimizing memory

consumption. The scenarios include peak usage, sudden data fluctuations, intensive computational processing, and burst data transmissions.

Under peak usage conditions, the original memory consumption of 500 KB is efficiently reduced to 350 KB, resulting in a 30% reduction. This reduction enhances the system's ability to manage heightened demand without compromising responsiveness.

In scenarios where environmental parameters exhibit sudden fluctuations, the system adapts. The original memory consumption of 600 KB experiences a 30% reduction, bringing it down to 420 KB. This demonstrates the resilience of memory reduction strategies in handling abrupt changes in data patterns.

During periods of intensive computational processing, the system showcases its ability to optimize memory usage. The original consumption of 700 KB is effectively reduced to 490 KB, emphasizing the versatility of memory reduction techniques in scenarios demanding elevated computational resources.

In situations requiring rapid data transmissions, the system excels in memory optimization. The original memory consumption of 550 KB undergoes a 30% reduction, reaching 385 KB. This underlines the significance of memory reduction strategies in maintaining efficiency during bursts of data activity.

The effectiveness showcased in Table 2 is attributed to targeted memory reduction techniques, specifically optimized data compression and intelligent flushing of non-essential data. These techniques prove instrumental in sustaining system responsiveness, particularly when confronted with abrupt changes in environmental parameters.

The optimized data compression technique involves compressing data in a way that reduces its size while preserving essential information. By implementing optimized data compression, the system minimizes the memory footprint of stored information, ensuring efficient utilization of resources during peak demands and sudden data fluctuations.

The system employs intelligent algorithms to identify and flush non-essential data, prioritizing critical information for real-time processing. This targeted flushing of unnecessary data ensures that the memory space is utilized for relevant and timely information, contributing significantly to the system's adaptability during scenarios involving burst data transmissions or computational intensity.

In essence, the combination of optimized data compression and intelligent flushing of non-essential data reflects a proactive approach to memory management. These techniques not only optimize memory consumption but also enhance the system's ability to navigate and respond effectively to dynamic environmental conditions, aligning seamlessly with the goals of environmental health monitoring applications.

Our study emphasizes that efficient memory management functions for ensuring system scalability. The adaptive allocation of resources positions the IoT system to seamlessly integrate additional sensors, thus expanding its capabilities organically without compromising performance, as in Table 3.

Table 3. System Scalability and Resource Adaptation

Number of Sensors	Original System Capacity	Scalability Achieved	Key Observations and Insights
5	Low	Achieved	Even with a minimal number of devices, the system demonstrated successful scalability, showcasing adaptability to a small-scale deployment.
10	Moderate	Achieved	The system continued to scale efficiently as devices increased, indicating versatility in handling a growing network.
15	Moderate-High	Achieved	With a slight rise in devices, the system maintained high performance, highlighting its robust scalability and resource allocation capabilities.
20	High	Achieved	The system consistently adapted to increased device counts, affirming its capability to handle a diverse and growing network.

The system showcased scalability even with a minimal number of devices, proving its adaptability to smaller-scale deployments. As the number of devices increased, the system demonstrated scalability, ensuring that the architecture could handle growing workloads effectively. The system efficiently allocated

resources even at lower device counts, indicating a balanced approach to resource management. The ability to scale down effectively suggests that the system is versatile and can be adapted for deployments with varying device requirements, providing flexibility in environmental monitoring applications.

CONCLUSIONS

In conclusion, our exploration into the realm of IoT systems and environmental health has unravelled the dynamics of memory management, unveiling its role in ensuring the optimal functionality and adaptability of the system. The paper outlined an approach to design and set up the IoT system, utilizing Python as the backend and Home Assistant Community Store (HACS) for the graphical user interface.

The memory management strategies discussed, such as memory allocation breakdown, memory reduction techniques, and system scalability assessments, collectively contribute to the robustness of the IoT system. The optimized data compression and flushing of non-essential data emerged as instrumental techniques in maintaining responsiveness, particularly in the face of changes in environmental parameters.

The practical implementation of the IoT system provides insights into its user-friendly interface, mobile application, and various integrations with protocols and devices. The system's ability to create zones, manage devices, and offer detailed reports provides support for environmental health monitoring. The presented results and discussions underscore the successful application of memory management strategies, making the IoT system adaptive, scalable, and resource efficient.

Looking ahead, future research in the domain of memory management for IoT systems in environmental health monitoring holds space for exploration and enhancement. Optimizing memory reduction strategies by delving into advanced compression algorithms and sophisticated intelligent flushing mechanisms presents an opportunity for more efficient memory utilization in resource constrained IoT devices. Additionally, the integration of edge computing could be investigated to reduce latency and enhance real-time responsiveness. Implementing machine learning algorithms for dynamic memory allocation based on changing environmental conditions and user preferences is another area that could enhance adaptability. Strengthening security measures through advanced encryption techniques and secure communication protocols is important for applications related to environmental health. Exploring the scalability of the IoT system in larger deployments, user-centric enhancements for a more intuitive interface, and integration with wearable devices for continuous health monitoring represent key directions for future work. Furthermore, the exploration of new environmental sensors and their seamless integration into the IoT system, as well as investigating personalized features and feedback mechanisms, will contribute to the ongoing evolution of IoT ecosystems in environmental health monitoring.

REFERENCES

Abid, M.A., Afaqui, N., Khan, M.A., Akhtar, M.W., Malik, A.W., Munir, A., Ahmad, J., & Shabir, B. (2022). Evolution towards Smart and Software-Defined Internet of Things. *AI, 3*(1), 100-123.

Abu Bakar, R., & Kijsirikul, B. (2023). Enhancing Network Visibility and Security with Advanced Port Scanning Techniques. *Sensors, 23*(17), 1-27.

Ajani, T. S., Imoize, A. L., & Atayero, A. A. (2021). An Overview of Machine Learning within Embedded and Mobile Devices–Optimizations and Applications. *Sensors, 21*(13), 1-44.

Akhigbe, B.I., Munir, K., Akinade, O., Akanbi, L., & Oyedele, L.O. (2021). IoT Technologies for Livestock Management: A Review of Present Status, Opportunities, and Future Trends. *Big Data and Cognitive Computing, 5*(1), 10.

Alhaidari, F. A., & Alqahtani, E. J. (2020). Securing Communication between Fog Computing and IoT Using Constrained Application Protocol (CoAP): A Survey. *Journal of Communications, 15*(1), 14-30.

Amanlou, S., Hasan, M. K., & Bakar, K. A. A. (2021). Lightweight and Secure Authentication Scheme for IoT Network based on Publish–Subscribe Fog Computing Model. *Computer Networks, 199*, 1-8.

Bansal, S., & Kumar, D. (2020). IoT Ecosystem: A Survey on Devices, Gateways, Operating Systems, Middleware and Communication. *International Journal of Wireless Information Networks, 27*, 340-364.

Chen, T., Wang, M., Su, J., Ikram, R. M. A., & Li, J. (2023). Application of Internet of Things (IoT) Technologies in Green Stormwater Infrastructure (GSI): A Bibliometric Review. *Sustainability, 15*(18), 1-22.

Concha Salor, L., & Monzon Baeza, V. (2023). Harnessing the Potential of Emerging Technologies to Break down Barriers in Tactical Communications. *Telecom, 4*(4), 709-731.

Donta, P. K., Srirama, S. N., Amgoth, T., & Annavarapu, C. S. R. (2022). Survey on Recent Advances in IoT Application Layer Protocols and Machine Learning Scope for Research Directions. *Digital Communications and Networks, 8*(5), 727-744.

Elgazzar, K., Khalil, H., Alghamdi, T., Badr, A., Abdelkader, G., Elewah, A., & Buyya, R. (2022). Revisiting the Internet of Things: New Trends, Opportunities and Grand Challenges. *Frontiers in the Internet of Things, 1*, 1-18.

Fortino, G., Guerrieri, A., Pace, P., Savaglio, C., & Spezzano, G. (2022). IoT Platforms and Security: An Analysis of the Leading Industrial/Commercial Solutions. *Sensors*, 2(6), 1-17.

Heidari, A., & Jabraeil Jamali, M. A. (2022). Internet of Things Intrusion Detection Systems: A Comprehensive Review and Future Directions. *Cluster Computing*, 1-28.

Imteaj, A., Thakker, U., Wang, S., Li, J., & Amini, M. H. (2021). A Survey on Federated Learning for Resource-Constrained IoT Devices. *IEEE Internet of Things Journal, 9*(1), 1-24.

Kondoro, A., Dhaou, I. B., Tenhunen, H., & Mvungi, N. (2021). Real Time Performance Analysis of Secure IoT Protocols for Microgrid Communication. *Future Generation Computer Systems, 116*, 1-12.

Ma, Z., Xiao, M., Xiao, Y., Pang, Z., Poor, H. V., & Vucetic, B. (2019). High-Reliability and Low-Latency Wireless Communication for Internet of Things: Challenges, Fundamentals, and Enabling Technologies. *IEEE Internet of Things Journal, 6*(5), 7946-7970.

Mniszewski, S. M., Belak, J., Fattebert, J. L., Negre, C. F., Slattery, S. R., Adedoyin, A. A., Bird, R. F., Chang, C., Chen, G., Ethier, S., & Fogerty, S. (2021). Enabling Particle Applications for Exascale Computing Platforms. *The International Journal of High Performance Computing Applications, 35*(6), 572-597.

Samaila, M. G., Sequeiros, J. B., Simoes, T., Freire, M. M., & Inacio, P. R. (2020). IoT-HarPSecA: A Framework and Roadmap for Secure Design and Development of Devices and Applications in the IoT Space. *IEEE Access, 8*, 16462-16494.

Saqib, E., Leal, I. S., Shallari, I., Jantsch, A., Krug, S., & O'Nils, M. (2023). Optimizing the IoT Performance: A Case Study on Pruning a Distributed CNN. 2023 IEEE Sensors Applications Symposium (SAS), 1-6.

Silva, D., Carvalho, L. I., Soares, J., & Sofia, R. C. (2021). A Performance Analysis of Internet of Things Networking Protocols: Evaluating MQTT, CoAP, OPC UA. *Applied Sciences, 11*(11), 4879.

Tsigkanos, C., Nastic, S., & Dustdar, S. (2019). Towards Resilient Internet of Things: Vision, Challenges, and Research Roadmap. *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 1754-1764.