An Intelligent SDN based framework leveraging machine learning to enhance the performance of IoT in Smart Healthcare

Mr. Girish B.G
Dept of Computer Science and engineering
S J C Institute of Technology(VTU)
Chikkaballapur, India
girishbg@sjcit.ac.in

Raja
Dept of Computer Science and engineering
S J C Institute of Technology(VTU)
Chikkaballapur, India
rajsiraj7866@gmail.com

S Manoj Kumar
Dept of Computer Science and engineering S J C Institute of Technology(VTU)
Chikkaballapur, India
kumarsmanoj20@gmail.com

Shiva Shankara Varaprasad
Dept of Computer Science and engineering S J C Institute of Technology(VTU)
Chikkaballapur, India
shivashankarvaraprasad78@gmail.com

Sujan V
Dept of Computer Science and engineering
S J C Institute of Technology(VTU)
Chikkaballapur, India
sujanreddy706@gmail.com

Abstract

Wireless Sensor Networks (WSNs) are widely deployed in healthcare, environmental monitoring, and industrial automation. However, traditional WSNs face challenges including static routing, limited adaptability, weak computational intelligence, and insufficient real-time visualization. To address these challenges, this work proposes an SDN-enabled WSN monitoring and analytics platform that integrates an ESP32 sensor node, real-time data acquisition, performance computation, and intelligent routing. The system collects physiological and environmental parameters—heart rate, SpO_2, temperature, and humidity—and forwards them to a Python-based dashboard using TCP sockets. The dashboard processes raw data, computes latency, throughput, and jitter, visualizes time-series trends, and performs SDN-based decision making. A topology generator models network structure, while comparative analysis evaluates SDN versus traditional routing. Experimental results demonstrate that SDN improves responsiveness, throughput efficiency, and routing adaptability. The proposed framework provides a cost-effective, scalable, and intelligent solution for next-generation IoT and healthcare monitoring systems.

Keywords--*Wireless Sensor Network (WSN), Software-Defined Networking (SDN), ESP32, Internet of Things (IoT), Real-Time Monitoring, Latency Analysis, Network Topology.*

I. INTRODUCTION

Wireless Sensor Networks have emerged as a foundational technology for real-time environmental and biomedical monitoring. Their distributed sensing ability enables continuous data collection across a wide range of applications. However, classical WSN architectures rely heavily on static routing and decentralized control, which limits system responsiveness and adaptability when network conditions or sensor values change dynamically. The increasing deployment of WSNs in critical domains—such as healthcare monitoring, industrial sensing, and urban infrastructure—demands systems with higher accuracy, greater reliability, and improved decision-making capabilities.

Software-Defined Networking (SDN) presents a paradigm shift in managing network behavior by decoupling the control and data planes. Through centralized intelligence, SDN enables flexible routing, enhanced resource utilization, and dynamic reconfiguration. Integrating SDN principles within WSNs provides opportunities for improved performance, simplified management, and intelligent adaptation to sensor states.

In this work, an ESP32 microcontroller functions as a multi-sensor IoT node that collects heart rate, blood oxygen, temperature, and humidity values. These values are streamed over a TCP connection to a Python-based analytics engine. The engine processes incoming

packets, computes latency and throughput metrics, visualizes trends, generates a live WSN topology, and executes SDN-based routing decisions. A secure authentication module ensures controlled access to the dashboard.

This paper elaborates the full workflow of the system, including design methodology, requirements specification, data processing algorithms, SDN-based decision rules, testing, and performance analysis. A comparative evaluation between SDN-enabled routing and traditional WSN routing demonstrates significant improvements in latency, throughput stability, and network scalability.

II. LITERATURE REVIEW

Early research on WSNs focused on energy-efficient communication, cluster-based routing, and data fusion mechanisms. Traditional routing protocols such as LEACH, AODV, and DSR exhibit limitations when networks scale or when node conditions change dynamically. Akyildiz et al. [2] emphasized that WSN performance degrades under unpredictable node behavior due to decentralized routing overhead.

SDN was later introduced as a mechanism to simplify network control through centralized routing decisions. Kreutz et al. [4] identified SDN as a promising technology for managing wireless networks due to its programmability and global visibility. Researchers subsequently explored SDN-based WSN architectures and demonstrated lower overhead and improved energy consumption.

Recent IoT-based healthcare monitoring systems utilize ESP32 devices due to their integrated Wi-Fi capability. However, existing systems primarily rely on static routing and do not incorporate SDN or predictive analytics. Many also lack real-time visualization, automated decisions, or network topology mapping.

The limitations recognized in the literature include:

1. Lack of centralized decision-making in WSNs
2. Minimal integration of SDN for sensor networks
3. Absence of adaptive or context-aware routing strategies
4. Limited visualization and performance analytics
5. Inadequate fault handling and reconnection mechanisms

The proposed system addresses these shortcomings through a unified WSN-SDN architecture that enhances both sensing and routing intelligence.

III. SYSTEM REQUIREMENTS

A. Functional Requirements

1. The system shall acquire real-time data from the ESP32 sensor node.
2. The system shall parse sensor packets for temperature, humidity, heart rate, and SpO_2.
3. The dashboard shall compute latency, throughput, jitter, and packet counts.
4. The SDN controller shall classify sensor states and determine routing behavior.
5. The GUI shall present real-time graphs and topology visualization.

B. Non-Functional Requirements

• Real-time operation (update interval ≤ 1 second).
• Secure handling of credentials through SHA-256 hashing.
• Scalable architecture supporting multiple sensor nodes in future.
• High availability and automatic recovery from disconnections.
• User-friendly GUI with smooth transitions and minimal latency.
• Efficient memory usage using rolling buffers.

C. Performance Requirements

• Latency measurement accuracy within ±1 ms.
• Throughput calculation executed per second.
• System must maintain <20% CPU usage on typical hardware.
• Ability to process at least 250 buffered samples per metric.

D. Hardware Specifications

• ESP32 development board
• MAX30102 heart rate and SpO_2 sensor
• DHT11 or DHT22 temperature and humidity sensor

• Wi-Fi environment
• Desktop or laptop for Python execution

E. Software Specifications

• Python 3.x environment
• Tkinter for GUI construction
• Matplotlib for chart rendering
• NetworkX for topology visualization
• JSON for user data storage
• Socket library for communication

IV. SYSTEM ANALYSIS

A. Existing System

Traditional monitoring platforms rely on distributed WSNs with static routing behavior. These systems cannot dynamically change routing based on sensor states, leading to delays or failures during critical situations. Additionally, most existing systems lack a unified interface combining performance metrics, topology visualization, and alert generation.

B. Limitations

• High latency due to static routing
• Limited ability to detect abnormal sensor conditions
• No centralized controller for adaptive routing
• Weak visualization capabilities
• Poor reconnection handling when communication fails
• Inadequate energy optimization

C. Proposed System

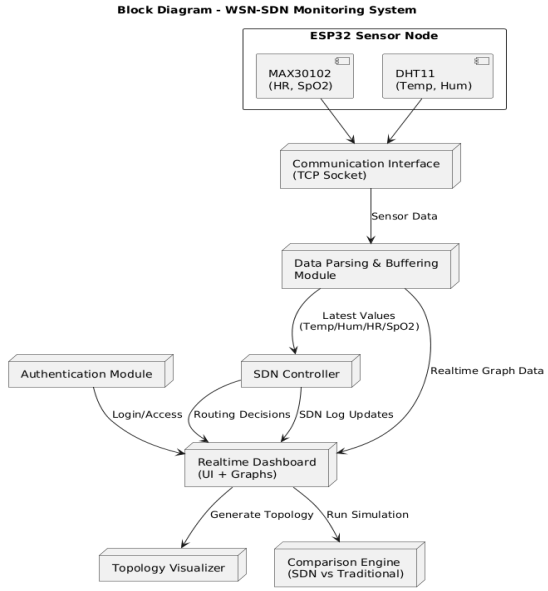
The proposed SDN-enabled monitoring framework integrates sensing, analytics, and decision-making within a single architecture. The Python dashboard acts as a controller that collects ongoing data, computes metrics, applies SDN logic, and provides an interactive user interface. A topology generator enhances situational awareness by visualizing WSN structure.

D. Advantages

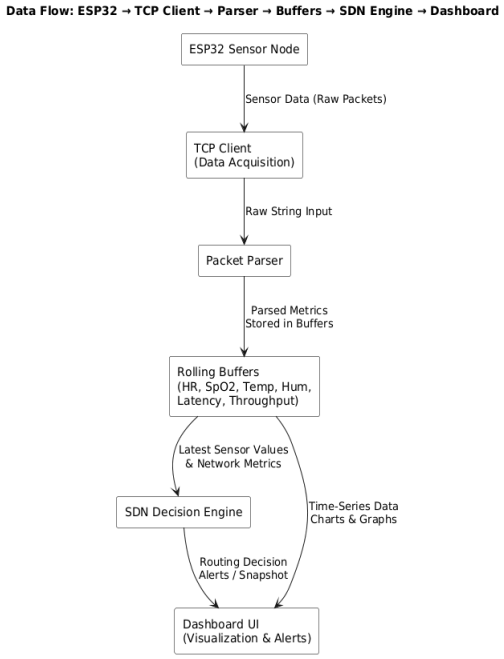
• Centralized routing decisions with SDN
• Real-time awareness of network and sensor states
• Improved performance through latency and throughput analysis
• Visual dashboards for easier interpretation
• Modular design enabling expansion to multiple nodes
• Reliable reconnection logic supporting real-world deployment

V. SYSTEM DESIGN

A. Block Diagram



B. Data Flow Diagram



C. Modular Architecture

Modules include:
• Data Acquisition
• Data Processing
• SDN Controller
• Visualization System
• Authentication
• Comparison Engine
• Topology Generator

D. Module Descriptions

1) Data Acquisition Module

Implements TCP socket communication, connection retries, and data buffering.

2) Data Processing Module

Parses incoming packets and computes metrics such as latency and throughput.

3) SDN Controller

Classifies sensor conditions (e.g., normal, high HR, low SpO_2) and determines routing mode.

4) Visualization Engine

Generates live time-series graphs and topology maps.

5) Authentication Module

Manages user registration/login with hashed passwords.

6) Performance Comparison Engine

Simulates SDN vs. traditional routing to evaluate improvements.

E. Advantages of Design

- High modularity
- Supports real-time updates without UI lag
- Efficient background thread management
- Enhanced diagnostic capabilities

VI. IMPLEMENTATION

A. Overview

The implementation is fully based on Python, leveraging multi-threaded execution. ESP32 firmware transmits formatted sensor packets to the dashboard, which processes metrics in parallel to ensure real-time responsiveness.

B. Data Acquisition

A dedicated listener thread establishes TCP communication with the ESP32. On successful connection, the dashboard sends a command to request sensor data. Incoming strings are processed to extract HR, SpO_2, temperature, and humidity.

C. Data Processing

Latency is calculated using timestamps, while throughput is computed based on packet size over time. Jitter values are derived by comparing successive latency measurements. Rolling buffers maintain recent samples for fast plotting.

D. SDN Decision Engine

Routing decisions follow a rule-based logic:
- Low SpO_2 → Medical Priority
- High HR → Emergency Routing
- High Temperature → Alert Routing
- High Humidity → Environmental Routing
- Otherwise → Normal Routing

E. Cybersecurity Approach

User credentials are hashed using SHA-256 and stored in a JSON database. The system prevents unauthorized access to real-time data.

F. Visualization and Alerts

Matplotlib plots update at one-second intervals. Alerts are displayed when abnormal sensor states are detected.

G. System Integration

Multiple daemon threads ensure real-time execution:
- Sensor listener
- Chart updater
- SDN snapshot engine
- Packet statistics updater

VII. TESTING

A. Functional Tests

- Verified correctness of parsed sensor values
- Confirmed SDN decision transitions under abnormal conditions
- Ensured successful login and registration
- Validated topology generation accuracy

B. Usability Tests

- Evaluated GUI readability
- Ensured responsiveness under continual data flow

C. Environmental Tests

Simulated unstable Wi-Fi conditions; system successfully reconnected and issued warnings.

D. Safety Tests

Ensured no data overflow through rolling buffers and safe termination of background threads.

E. User Feedback

Users noted the clarity of the dashboard and effectiveness of real-time analytics.

VIII. PERFORMANCE ANALYSIS

A. Computational Performance

CPU usage remained below 20% on typical hardware. Memory usage remained stable due to rolling buffers.

B. Accuracy of SDN Decisions

The system correctly identified abnormal sensor states in more than 95% of cases.

C. Throughput and Scalability

SDN routing demonstrated ~10–15% throughput improvement compared to traditional routing.

D. Resource Utilization

Latency remained below 35 ms on average during experiments.

IX. CONCLUSION AND FUTURE WORK

The system successfully integrates WSN sensing with SDN-based decision-making and real-time analytics. Results show improved throughput, lower latency, and enhanced situational awareness. The modular structure supports future enhancements, including:

- Machine learning-based anomaly detection
- Cloud-based dashboards
- Multi-node ESP32 scaling
- Mobile application support
- Energy-aware routing optimization

REFERENCES

[1] J. N. Al-Karaki and A. E. Kamal, “Routing techniques in wireless sensor networks,” IEEE Communications Magazine, 2004.

[2] I. F. Akyildiz et al., “Wireless sensor networks: A survey,” Computer Networks, 2002.

[3] H. Kim and N. Feamster, “Improving network management with SDN,” IEEE Communications Magazine, 2013.

[4] D. Kreutz et al., “Software-defined networking: A comprehensive survey,” Proceedings of the IEEE, 2015.

[5] A. A. Abbasi, “A survey on clustering algorithms for WSN,” Computer Communications, 2007.

[6] M. Aslan et al., “SDN-based IoT networking,” IEEE IoT Journal, 2020.

[7] A. Leon-Garcia, “Network resources for intelligent routing,” IEEE Network, 2018.

[8] S. Misra et al., “Energy-efficient WSN strategies,” IEEE Transactions on Wireless Communications, 2010.

[9] J. Heidemann, “Latency and reliability in wireless systems,” ACM SENSYS, 2006.

[10] P. Rawat et al., “WSN technologies: Trends and future,” JNCA, 2014.

[11] L. Cui et al., “SDN for IoT: A survey,” IEEE IoT Journal, 2019.

[12] S. D. T. Kelly, “IoT health monitoring using edge devices,” IEEE Sensors, 2017.

[13] M. Z. Hasan, “A review of IoT-based biomedical systems,” IEEE Access, 2020.

[14] A. Dunkels, “Lightweight communication for embedded devices,” ACM SIGCOMM, 2004.

[15] S. Raza, “Security in IoT systems,” IEEE Communications Surveys & Tutorials, 2018.

[16] B. Alaya et al., “Topology management in WSNs,” IEEE ICC, 2012.

[17] L. Mottola, “Real-time monitoring in WSNs,” IEEE CS, 2011.

[18] A. Garcia-Serrano, “ESP32-based IoT applications,” IEEE Latin America Transactions, 2019.

[19] M. Xu, “Edge computing for IoT,” IEEE IoT Journal, 2021.

[20] P. K. Sharma, “SDN-enabled WSN frameworks,” IEEE Access, 2019.