International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.
© 2026 Scholar AI LLC.
https://ijpsat.org/
Vol. 59 No. 1 September 2026, pp. 952-967

A Resilient Edge-Fog-Cloud IoT Architecture For Infant Health Monitoring In Resource-Constrained Community Settings: The IoT-CKS Model

Christophe KUNGUNIKO SOLA[1], José Emmanuel MATA TOMBO[2]

[1]Department of Physics, Faculty of Sciences and Technology, Institut Supérieur Pédagogique, Mbanza-Ngungu, Democratic Republic of the Congo

[2]Department of Physics and Applied Sciences, Faculty of Sciences, Université Pédagogique Nationale, Kinshasa, Democratic Republic of the Congo

Corresponding author: Christophe KUNGUNIKO SOLA, chrisstophekungunikosola@gmail.com

(cc) BY

Abstract: Background: Infant health monitoring in resource-constrained community settings is challenged by intermittent connectivity, unstable power supply, and limited technical resources. Architectures that depend predominantly on remote Cloud services may therefore lose important functions during communication disruptions. Methods: This study used a Design Science Research approach to develop IoT-CKS, a requirements-driven Edge-Fog-Cloud architecture for infant health monitoring, using Mbanza-Ngungu in the Democratic Republic of the Congo as the reference context. Contextual constraints were translated into system requirements, design principles, functional allocation, architectural models, and implementation-oriented technology choices. Results: The resulting architecture distributes physiological data acquisition, preprocessing, local evaluation of predefined abnormal measurement conditions, immediate local alerting, and temporary storage to the Edge layer; aggregation, messaging, communication management, and delayed synchronization to the Fog layer; and persistent storage, broader analysis, visualization, supervision, and administration to the Cloud layer. The design explicitly supports degraded local operation during temporary connectivity loss and recovery through delayed synchronization after reconnection. Conclusion: The contribution of IoT-CKS lies not in the individual Edge, Fog, Cloud, MQTT, or sensing technologies, but in their traceable, context-aware integration into a resilience-oriented architecture. The model provides a structured foundation for subsequent prototype implementation, technical evaluation, field assessment, and eventual clinical validation.

Keywords: Design Science Research; distributed computing; service continuity; local autonomy; delayed synchronization; biomedical sensing; offline operation; community healthcare.

1. Introduction

The Internet of Things (IoT) has substantially transformed the way physical devices, communication networks, computing resources, and data-driven services interact. IoT systems enable distributed sensing devices to acquire, exchange, and process information across interconnected environments, supporting applications in industry, agriculture, transportation, smart environments, and healthcare [1], [2].

Healthcare represents a particularly important application domain because connected sensing technologies can support continuous or periodic acquisition of physiological measurements and facilitate remote monitoring services [3], [4]. The Internet of Medical Things has consequently emerged as an important extension of IoT, linking biomedical sensors, embedded devices, communication networks, and computing platforms to healthcare applications.

Recent research also shows an increasing use of Edge, Fog, and Cloud resources in IoMT architectures to address latency, distributed processing, resource utilization, and service responsiveness [5], [6], [7], [8]. However, the effectiveness of such architectures depends strongly on the conditions of the intended deployment environment.

Many IoT healthcare systems are designed under assumptions of sufficiently reliable network connectivity, available electrical power, and access to technical infrastructure. These assumptions cannot always be maintained in resource-constrained community environments. Temporary network interruptions, unstable electrical supply, restricted financial resources, and limited availability of specialized technical personnel may compromise systems whose critical functions rely predominantly on remote computing infrastructure.

This challenge is particularly relevant to infant health monitoring, where the system should continue to acquire and process physiological measurements even when access to remote services is temporarily unavailable. The architectural problem is therefore not limited to connecting biomedical sensors to a Cloud platform. It also concerns where sensing, processing, storage, communication, alerting, and synchronization functions should be located so that essential monitoring services can continue during infrastructure disruption.

Edge and Fog Computing provide useful architectural mechanisms for addressing this problem. Fog Computing extends computing, communication, and storage capabilities between end devices and centralized Cloud resources [9], while Edge Computing places selected computational functions close to the sources of data [10], [11], [12]. In healthcare-oriented architectures, these paradigms can complement Cloud platforms by enabling selected processing and decision-support functions to occur closer to sensing devices [6], [7].

The present study addresses this architectural challenge through the development of IoT-CKS, a distributed Edge-Fog-Cloud model intended for infant health monitoring in resource-constrained community settings. Mbanza-Ngungu, in the Democratic Republic of the Congo, serves as the reference context for identifying the operational constraints and requirements considered during the design process. The underlying research explicitly treats intermittent connectivity, power limitations, constrained technical resources, and service continuity as design considerations rather than secondary implementation issues.

IoT-CKS does not seek to introduce a new biomedical sensor, communication protocol, or distributed-computing paradigm. Instead, its contribution lies in the systematic translation of contextual constraints into requirements, functional allocation, architectural responsibilities, and resilience mechanisms.

The study therefore addresses the following research question: How can a distributed Edge-Fog-Cloud IoT architecture be designed to support resilient infant health monitoring under intermittent connectivity, power limitations, and constrained technical resources in community settings?

The study makes three principal contributions. First, it establishes a traceable relationship between contextual constraints and system requirements. Second, it proposes a distributed Edge-Fog-Cloud architecture in which local processing, temporary storage, local alert generation, Fog-based coordination, and delayed synchronization are explicitly integrated as resilience mechanisms. Third, it provides a structured mapping between requirements, architectural functions, and implementation-oriented technologies.

The architectural contribution presented in this article is intentionally separated from the detailed experimental performance analysis of IoT-CKS. The present paper focuses on what was designed, why it was designed in this way, and how the architecture responds to identified contextual requirements. Detailed performance and statistical validation are treated as a separate research contribution.

The remainder of the paper is organized as follows. Section 2 reviews related work. Section 3 describes the DSR-based methodology and requirements-engineering process. Section 4 presents the proposed architecture. Section 5 describes system modeling and technology selection. Section 6 discusses the contribution and positioning. Section 7 presents limitations and future work, and Section 8 concludes the study.

2. Related Work

2.1 IoT and IoMT Architectures for Health Monitoring

The Internet of Things has progressively expanded the capabilities of healthcare information systems by enabling interconnected sensing devices to acquire, transmit, and process physiological and contextual data. Early IoT research established the architectural foundations of interconnected smart objects and distributed services [1], [2], while subsequent work demonstrated their potential for healthcare monitoring and medical applications [3].

The emergence of the Internet of Medical Things (IoMT) further extended this concept by integrating biomedical sensors, embedded systems, communication technologies, data-processing platforms, and healthcare applications. Reference [4] highlighted the potential of IoMT technologies for monitoring, remote healthcare services, and medical data management. More recent systematic reviews confirm continued development while identifying challenges related to communication reliability, interoperability, scalability, resource management, security, and efficient data processing [5], [8].

For infant health monitoring in resource-constrained community environments, the architectural challenge extends beyond physiological sensing. The system must determine which functions should remain available locally, which can be delegated to intermediate computing resources, and which can safely depend on remote services.

2.2 From Cloud-Centered Processing to Edge-Fog-Cloud Computing

Cloud Computing provides substantial storage and computational resources and has played an important role in IoT applications. Nevertheless, architectures that place most processing and decision-support functions in remote Cloud infrastructure may become dependent on communication availability and may introduce additional latency and bandwidth requirements.

Fog Computing extends computing, networking, and storage capabilities closer to end devices and data sources [9]. Edge Computing emphasizes processing at or near the network edge, allowing selected operations to occur close to the devices generating the data [10], [11], [12]. Recent healthcare-oriented work has explored Fog- and Edge-assisted architectures, including Fog-Cloud healthcare monitoring [7] and Edge-Fog-Cloud optimization for IoMT [6].

These developments demonstrate that the use of Edge and Fog resources in healthcare is no longer itself a sufficient basis for claiming architectural novelty. The relevant research question is increasingly concerned with how responsibilities are distributed, why particular functions are assigned to particular layers, and how that distribution responds to the operational requirements of the target environment.

2.3 Resilience and Service Continuity under Infrastructure Constraints

Resource-constrained environments introduce requirements that are not adequately represented by performance optimization alone. A healthcare monitoring architecture may achieve low latency under normal connectivity while still becoming functionally unavailable when communication with remote infrastructure is interrupted.

In IoT-CKS, resilience is considered at the architectural level. The objective is not to eliminate failures or guarantee uninterrupted availability of every component. Instead, the architecture seeks to preserve essential local monitoring functions during temporary infrastructure disruption and to recover deferred communication functions after connectivity is restored.

This behavioral perspective implies three operational states: normal connected operation, temporary degraded/offline operation, and recovery/synchronization following restoration of communication.

2.4 Healthcare IoT in Resource-Constrained Community Settings

Resource-constrained community environments may combine intermittent network availability, electrical instability, restricted budgets, geographically dispersed monitoring points, and limited access to specialized technical personnel. Architecture design therefore requires affordability, maintainability, energy consumption, local autonomy, communication efficiency, and service continuity to be considered together.

In IoT-CKS, resource constraints are incorporated during requirements identification and architectural modeling rather than treated only as deployment problems. Technology-selection criteria include cost, availability, energy consumption, integration capability, compatibility with distributed operation, scalability, and reproducibility.

2.5 Research Gap and Positioning of IoT-CKS

Recent research confirms that healthcare IoT architectures increasingly adopt distributed Edge, Fog, and Cloud resources to address latency, communication overhead, energy consumption, resource utilization, and service responsiveness [10], [11], [12], [6], [7]. Recent reviews likewise show that networking, scalability, resource management, security, interoperability, and distributed processing remain important challenges [5], [8].

The research gap addressed here is therefore not an absence of Edge-Fog-Cloud healthcare architectures. Instead, it concerns the explicit and traceable translation of resource-constrained community conditions into system requirements, functional allocation, offline behavior, temporary data retention, delayed synchronization, and architectural decisions for infant health monitoring.

IoT-CKS is positioned at this intersection. Its novelty does not reside in individual Edge, Fog, Cloud, MQTT, or biomedical sensing technologies, but in their requirements-driven integration into a context-aware and resilience-oriented architecture.

The conceptual chain underlying the model is: Contextual constraints -> System requirements -> Design principles -> Functional allocation -> Edge-Fog-Cloud architecture -> Technology selection -> IoT-CKS artefact.

3. Materials and Methods

3.1 Research Design

This study adopted Design Science Research (DSR) as the methodological framework for development of the IoT-CKS architecture. DSR is appropriate when an identified practical problem is addressed through systematic design of an artefact grounded in explicit requirements and scientific knowledge [13], [14], [15].

In this study, the artefact is the IoT-CKS architectural model. The purpose was not to develop a clinically validated medical device, but to construct a coherent and reproducible architectural model capable of translating contextual constraints into functional and technical design decisions.

The methodological process was organized as: Problem identification -> Context and requirements analysis -> Definition of design principles -> Architectural modeling -> Technology selection -> Preparation for technical evaluation. The process was iterative, with architectural decisions checked against their originating requirements.

3.2 Reference Context and Design Constraints

The architecture was developed with Mbanza-Ngungu, Democratic Republic of the Congo, as the reference community context. The requirements analysis considered intermittent network connectivity, limitations in electrical power availability, restricted financial resources, limited access to specialized technical personnel, geographic dispersion of monitoring points, and the need to preserve timely local monitoring functions. These constraints were treated as architectural inputs rather than postponed until implementation.

3.3 Requirements Identification and Analysis

Requirements identification was guided by requirements-engineering principles and by the DSR objective of translating an identified problem into explicit design requirements [13], [16]. Functional requirements include physiological data acquisition, local processing, anomaly detection, alert generation, data transmission, temporary storage, synchronization, visualization, and supervision. Contextual and non-functional requirements include resilience, affordability, energy-conscious operation, modularity, maintainability, scalability, and reproducibility.

Table 1. Translation of contextual constraints into IoT-CKS architectural requirements

Contextual constraint	Potential operational impact	Derived requirement	Architectural response
Intermittent connectivity	Remote services may become temporarily unavailable	Preserve essential local operation	Local processing, temporary storage, delayed synchronization
Power limitations	Monitoring functions may be interrupted	Reduce unnecessary resource dependence	Energy-conscious Edge operation and distributed processing
Limited financial resources	Large-scale deployment may become difficult	Affordability	Low-cost and modular components
Limited specialized technical resources	Maintenance may become difficult	Maintainability and simplicity	Modular architecture and replaceable components
Geographic dispersion	Centralized coordination may be difficult	Distributed coordination	Edge nodes and Fog gateway
Need for timely response	Remote communication may delay essential response	Local responsiveness	Local anomaly detection and immediate alerting

3.4 Derivation of Design Principles

The requirements were translated into four principal design principles: functional distribution, local autonomy, resilience, and modularity. Functional distribution determines where individual monitoring functions should execute. Local autonomy requires essential monitoring functions to remain executable when Cloud communication is temporarily unavailable. Resilience concerns preservation of essential functions during disruption and recovery of deferred communication after restoration. Modularity separates architectural functions from specific implementation technologies.

3.5 Architectural Modeling

The IoT-CKS architecture was formalized through complementary structural and behavioral representations based on standardized UML modeling concepts [17]. Structural modeling represents sensing components, Edge nodes, Fog resources, Cloud services, storage functions, and communication interfaces. Behavioral modeling represents interactions during normal operation and communication disruption.

3.6 Technology Selection Strategy

Technology selection was performed after functional responsibilities had been established. Selection criteria included affordability, availability, energy consumption, ease of integration, compatibility with distributed operation, reliability, scalability, maintainability, and reproducibility. An ESP32-class microcontroller was selected as a feasible Edge platform; MLX90614 and MAX30102 modules were considered for physiological sensing. MQTT and Eclipse Mosquitto were selected for distributed messaging and brokering. SQLite, InfluxDB, Node-RED, and Grafana were associated respectively with lightweight local storage, time-series persistence, flow orchestration, and visualization.

3.7 Requirements-to-Architecture Traceability

A central methodological objective was to preserve traceability. The resulting design chain is: Contextual constraint -> System requirement -> Design principle -> Functional allocation -> Architectural component -> Technology candidate. For example, intermittent connectivity leads to a service-continuity requirement, local autonomy, local processing and temporary storage, and implementation at the Edge/Fog levels.

3.8 Methodological Scope of the Present Article

The complete IoT-CKS research includes a subsequent technical evaluation stage using tools such as ns-3, Python, Eclipse Mosquitto, Node-RED, InfluxDB, and Grafana. Detailed experimental scenarios, repeated runs, performance indicators, statistical tests, and quantitative results are deliberately excluded from the present article. Article 1 addresses the design question; the complementary experimental study addresses how the resulting architecture behaves under controlled operational constraints.

4. Proposed IoT-CKS Architecture

4.1 Overall Architectural Design

IoT-CKS is designed as a distributed three-layer architecture comprising Edge, Fog, and Cloud computing resources. Responsibilities are assigned according to urgency, dependence on connectivity, computational requirements, and role in service continuity. Essential monitoring functions remain close to the physiological data source, whereas functions requiring broader aggregation, permanent storage, visualization, or centralized supervision are progressively delegated to Fog and Cloud resources.

Under normal conditions, the principal flow is Biomedical sensors -> Edge node -> Fog layer -> Cloud services. During temporary remote unavailability, the alternative path is Biomedical sensors -> Edge processing -> Local decision/alert -> Temporary storage -> Fog coordination -> Delayed Cloud synchronization.

Table 2. Functional distribution across the IoT-CKS architecture

Layer	Main components	Primary responsibilities	Role in resilience
Edge	Biomedical sensors, ESP32-class node, local memory	Acquisition, preprocessing, local anomaly detection, immediate alerting, temporary storage	Preserves essential local monitoring
Fog	Local gateway, MQTT broker, intermediate services/storage	Aggregation, coordination, communication management, buffering and synchronization	Maintains local coordination and manages deferred communication
Cloud	Remote server, persistent database, visualization and administration services	Long-term storage, broader analysis, visualization, supervision, administration	Restores centralized services after connectivity recovery

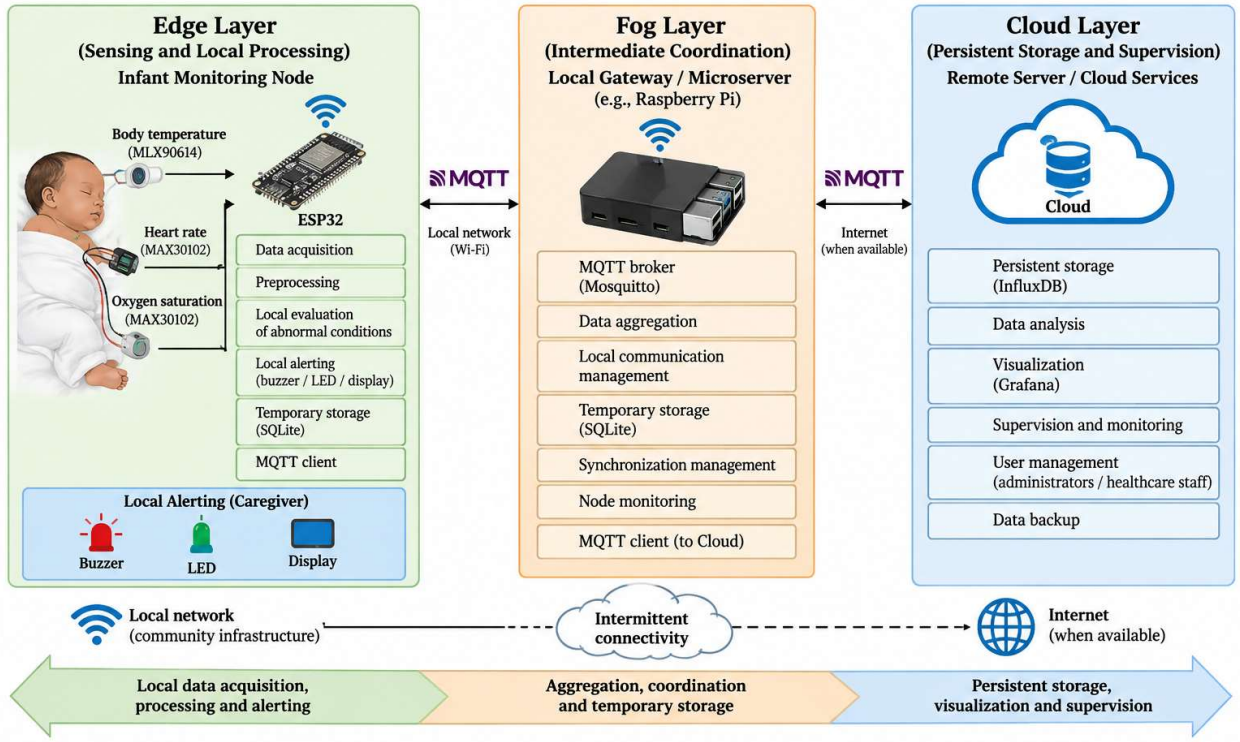

Figure 1. Proposed IoT-CKS Edge–Fog–Cloud Architecture for Infant Health Monitoring in Resource-Constrained Community Settings.

Figure 1. Proposed IoT-CKS Edge-Fog-Cloud Architecture for Infant Health Monitoring in Resource-Constrained Community Settings.

4.2 Edge Layer: Sensing and Local Autonomy

The Edge layer is located closest to the monitored child and combines physiological sensing with embedded processing. The implementation-oriented configuration associates MLX90614 with temperature acquisition and MAX30102 with SpO2 and heart-rate-related acquisition, connected to an ESP32-class platform. The Edge performs acquisition, preprocessing, local evaluation of predefined abnormal measurement conditions, immediate local alert generation when required, and temporary data retention. Local anomaly detection is not automated medical diagnosis; clinical interpretation remains the responsibility of qualified healthcare professionals.

4.3 Fog Layer: Local Coordination and Communication Management

The Fog layer forms the intermediary between Edge devices and remote Cloud infrastructure. It provides a local coordination point for message aggregation, communication management, temporary storage, local service coordination, and management of data awaiting synchronization. MQTT-based messaging and an Eclipse Mosquitto broker can support this role. The Fog layer reduces the need for every Edge node to interact continuously and independently with the Cloud.

4.4 Cloud Layer: Persistence, Supervision, and Global Services

The Cloud supports long-term data storage, broader analysis, visualization, system supervision, historical consultation, and administrative services. InfluxDB and Grafana are feasible candidates for time-series persistence and visualization, while Node-

RED may support orchestration. These software products are implementation candidates rather than intrinsic definitions of the architecture.

4.5 Normal Connected Operation

When connectivity is available, physiological measurements are acquired and preprocessed at the Edge, locally evaluated, coordinated through the Fog, and transferred to Cloud services for persistence and visualization. Even in normal mode, the Edge remains functionally relevant rather than acting as a passive transmitter.

4.6 Operation during Connectivity Interruption

When access to remote services becomes temporarily unavailable, IoT-CKS enters a degraded but locally functional state. Physiological acquisition, preprocessing, predefined local condition evaluation, local alerting, and temporary retention can continue. The architecture therefore distinguishes temporary loss of remote services from complete loss of monitoring functionality.

4.7 Connectivity Recovery and Delayed Synchronization

After communication is restored, pending measurements can be transferred through the Fog layer and synchronized with remote infrastructure. The resilience cycle is: Connected operation -> Connectivity interruption -> Local autonomous operation -> Temporary data retention -> Connectivity recovery -> Delayed synchronization -> Connected operation.

4.8 Requirements-to-Architecture Traceability

Table 3. Traceability between requirements and IoT-CKS architectural mechanisms

Identified requirement	Architectural decision	Functional mechanism	Expected architectural effect
Service continuity under intermittent connectivity	Distribute critical functions	Edge processing	Essential local functions remain available
Preserve unsent measurements	Provide local storage	Temporary data retention	Reduced risk of loss solely because remote communication is unavailable
Timely local response	Place condition evaluation near sensors	Edge anomaly detection and local alerting	No mandatory Cloud round trip for immediate local response
Distributed coordination	Introduce intermediate computing layer	Fog gateway and messaging services	Local aggregation and communication management
Recovery after interruption	Separate acquisition from immediate remote transmission	Delayed synchronization	Deferred data can be transmitted after reconnection
Long-term monitoring	Maintain remote persistent services	Cloud storage and visualization	Historical persistence and centralized supervision
Affordability and maintainability	Use modular and accessible components	Replaceable technology modules	Adaptability to resource constraints

4.9 Architectural Resilience Model

In IoT-CKS, resilience does not imply absence of failure. It refers to the architectural capacity to preserve essential local monitoring functions during temporary infrastructure unavailability and to recover deferred communication services when connectivity becomes available again. Four mechanisms jointly support this property: local autonomy, functional distribution, temporary data retention, and delayed synchronization.

4.10 Architectural Design Principle

The operating philosophy of IoT-CKS can be summarized as: Sense locally -> Process locally -> Decide locally when required -> Store temporarily when necessary -> Coordinate through Fog -> Synchronize when possible -> Persist and supervise in the Cloud.

5. System Modeling and Technology Selection

5.1 Modeling Strategy

The architecture was formalized through complementary structural and behavioral models. For the present article, three representations are retained: the overall architecture, deployment of functional components across the three layers, and the behavioral sequence describing offline operation and delayed synchronization.

5.2 Functional Deployment Model

At the Edge, biomedical sensors are associated with an embedded node responsible for acquisition, preprocessing, local condition evaluation, immediate alert generation, and temporary retention. At the Fog level, a local gateway and communication services coordinate exchanges and synchronization. At the Cloud level, persistent storage, broader analysis, visualization, supervision, and administrative services are provided.

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.
© 2026 Scholar AI LLC.
https://ijpsat.org/
Vol. 59 No. 1 September 2026, pp. 952-967

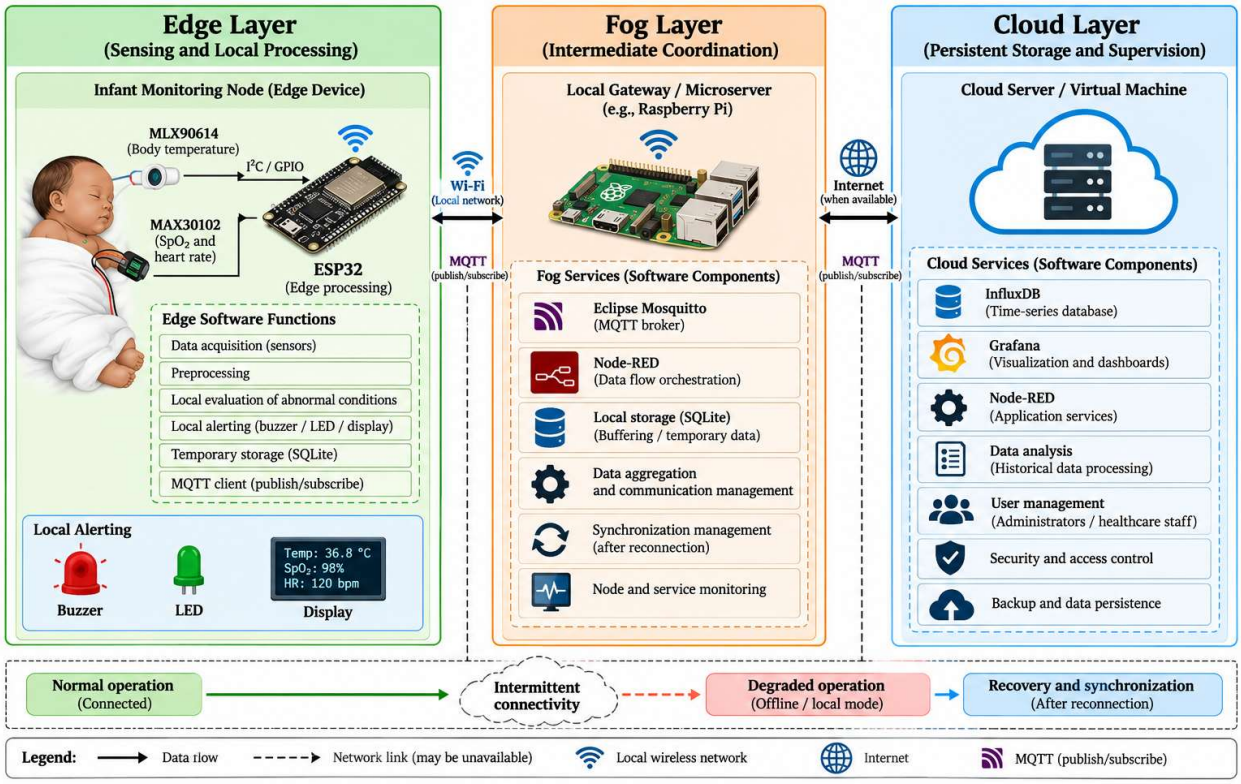

Figure 2. IoT-CKS Deployment Model across Edge, Fog, and Cloud Layers.

Figure 2. IoT-CKS Deployment Model across Edge, Fog, and Cloud Layers.

5.3 Behavioral Modeling of Connectivity Disruption

Under normal conditions, measurements progress from sensing through Edge and Fog resources toward Cloud persistence and visualization. When remote connectivity becomes unavailable, essential local functions continue and remote transmission is deferred. After reconnection, pending measurements are synchronized.

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.
© 2026 Scholar AI LLC.
https://ijpsat.org/
Vol. 59 No. 1 September 2026, pp. 952-967

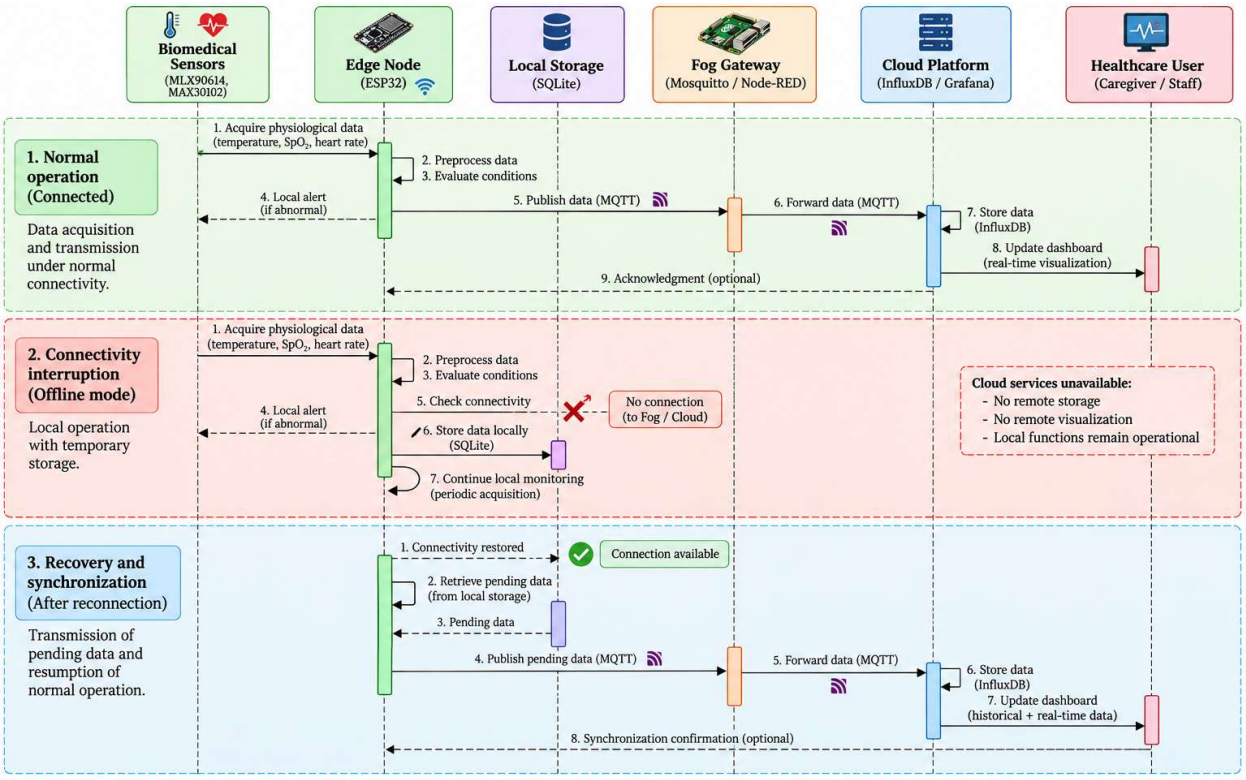

Figure 3. IoT-CKS Offline Operation and Delayed Synchronization Sequence.

Figure 3. IoT-CKS Offline Operation and Delayed Synchronization Sequence.

5.4 Edge Hardware and Biomedical Sensing Components

The implementation-oriented model associates the Edge layer with an ESP32-class platform and includes MLX90614 for temperature and MAX30102 for SpO2 and heart-rate-related acquisition. These components are feasible implementation candidates rather than mandatory elements of the architecture.

5.5 Communication and Messaging

MQTT was selected as the primary messaging protocol because its lightweight publish/subscribe communication model is compatible with constrained IoT environments and with the distributed organization adopted in IoT-CKS [18], [19]. Eclipse Mosquitto supports the broker function. The communication subsystem should nevertheless be distinguished from the architecture itself.

5.6 Data Storage, Orchestration, and Visualization

Data management distinguishes temporary local retention from persistent remote storage. SQLite is a lightweight option for local storage; InfluxDB supports time-series persistence; Grafana provides visualization; and Node-RED supports orchestration. Temporary storage supports resilience, whereas persistent storage supports historical data management and supervision.

5.7 Technology-to-Function Mapping

Table 4. Mapping of implementation technologies to IoT-CKS functions

Technology/component	Architectural layer	Main function	Architectural requirement addressed
MLX90614	Edge	Temperature acquisition	Physiological sensing
MAX30102	Edge	SpO2 and heart-rate-related acquisition	Physiological sensing
ESP32	Edge	Embedded processing and local control	Local autonomy
Local storage / SQLite	Edge/Fog	Temporary data retention	Continuity during communication interruption
MQTT	Edge-Fog-Cloud	Distributed messaging	Lightweight communication
Eclipse Mosquitto	Fog	Message brokering and coordination	Distributed communication management
Node-RED	Fog/Cloud	Data-flow orchestration	System integration
InfluxDB	Cloud/data layer	Time-series persistence	Long-term data management
Grafana	Cloud/application layer	Visualization and dashboarding	Monitoring and supervision

5.8 Architectural Modularity

Technology selection follows architectural design rather than defining it. The architecture specifies required functions and their distribution; the technology stack provides one feasible means of implementation. This separation improves adaptability as technologies, costs, availability, and deployment requirements change.

5.9 Reproducibility of the Design

Reproducibility means that another researcher should be able to reconstruct the reasoning from problem to artefact. The chain is: Context -> Constraints -> Requirements -> Design principles -> Functional allocation -> Architectural layers -> Technology-selection criteria -> Implementation candidates.

5.10 Consolidated System Model

Figure 1 answers what the architecture is; Figure 2 shows where components and functions are deployed; Figure 3 shows how the architecture behaves when connectivity is interrupted and restored. Together, these views establish the structural, deployment, and behavioral dimensions of the artefact without reproducing every diagram from the complete mémoire.

6. Discussion

6.1 Scientific Contribution of IoT-CKS

The scientific contribution of IoT-CKS should not be interpreted as introduction of a new Edge, Fog, Cloud, sensing, or messaging technology. These paradigms are established in IoT and IoMT literature [9], [3], [10], [11], [12]. Instead, IoT-CKS contributes a requirements-driven architectural integration in which allocation of system functions is explicitly derived from contextual and operational constraints.

6.2 Requirements-Driven Design

Intermittent connectivity is translated into requirements for local autonomy, temporary data retention, and delayed synchronization. Timely response motivates local processing and anomaly evaluation at the Edge. Distributed coordination motivates Fog resources, while long-term persistence and broader supervision are assigned to the Cloud. This traceability strengthens the explanatory and reproducible character of the artefact.

6.3 Context-Aware Functional Allocation

Existing literature establishes the advantages of bringing computation closer to data sources [9], [10], [11], and recent healthcare studies demonstrate the relevance of Fog- and Edge-assisted IoMT architectures [6], [7]. IoT-CKS applies these principles through context-specific functional allocation, distinguishing functions that can tolerate remote dependency from those that should remain locally executable.

6.4 Resilience by Design

Resilience in IoT-CKS does not imply uninterrupted availability of every service. Remote visualization may be unavailable during a network interruption, while physiological acquisition, local processing, local alerting, and temporary data retention remain operational. Thus, loss of Cloud connectivity is not equivalent to loss of all monitoring functions.

6.5 IoT-CKS versus a Predominantly Cloud-Centered Architecture

Table 5. Conceptual comparison between predominantly Cloud-centered monitoring and IoT-CKS

Design dimension	Predominantly Cloud-centered approach	IoT-CKS
Physiological acquisition	Local	Local
Immediate processing	May depend strongly on remote services	Primarily Edge-based for essential functions
Local anomaly evaluation	Not necessarily available	Explicitly allocated to Edge
Local alerting	May depend on remote processing	Can remain locally available
Temporary storage during disconnection	Architecture-dependent	Explicit resilience mechanism
Intermediate coordination	Optional	Fog layer
Long-term storage	Cloud	Cloud
Visualization/supervision	Primarily Cloud	Primarily Cloud
Operation during connectivity loss	Remote functions may be interrupted	Degraded local monitoring mode

Design dimension	Predominantly Cloud-centered approach	IoT-CKS
Recovery after reconnection	Implementation-dependent	Delayed synchronization explicitly modeled

This comparison is architectural rather than experimental. The present article does not claim that IoT-CKS experimentally outperforms every Cloud-centered architecture. The defensible conclusion is that IoT-CKS reduces architectural dependence on continuous Cloud connectivity for essential local functions.

6.6 Why the Fog Layer Is Retained

The Fog layer does not duplicate the Edge. The Edge is associated primarily with the individual sensing and monitoring point, whereas the Fog provides an intermediate coordination level capable of aggregating exchanges from Edge nodes, supporting messaging services, managing local communication, and participating in deferred synchronization. Its inclusion is justified by coordination scope.

6.7 Technology Independence and Modularity

ESP32, MLX90614, MAX30102, MQTT, Mosquitto, SQLite, Node-RED, InfluxDB, and Grafana constitute a feasible implementation-oriented configuration, but none individually constitutes the architecture. Functionally equivalent technologies can replace them while preserving the allocation of responsibilities.

6.8 Clinical Interpretation and Scope

Because IoT-CKS concerns infant health monitoring, the boundary between technical monitoring and clinical decision-making must be explicit. Local evaluation of predefined abnormal measurement conditions is not automated diagnosis. Clinical interpretation, diagnosis, treatment decisions, and patient management remain the responsibility of qualified healthcare professionals. IoT-CKS is a technical DSR artefact rather than a certified medical device.

6.9 Transferability beyond Mbanza-Ngungu

IoT-CKS was developed using Mbanza-Ngungu as the reference context. Its architectural reasoning may be transferable to other environments presenting comparable constraints, but transferability should not be assumed automatically. Deployment elsewhere requires renewed contextual and requirements analysis.

6.10 Positioning of the Contribution

The contribution can be positioned at three levels: requirements, architecture, and resilience. IoT-CKS provides a requirements-driven, context-aware Edge-Fog-Cloud architectural model in which essential infant health monitoring functions are deliberately distributed to preserve local operational continuity under temporary connectivity limitations while maintaining Fog-based coordination and Cloud-based persistent services.

7. Limitations and Future Work

IoT-CKS is presented primarily as a Design Science Research architectural artefact. Although the complete research includes a subsequent technical evaluation stage, the present study does not constitute a full-scale operational deployment in community healthcare facilities and should not be interpreted as a production-ready healthcare system.

The proposed sensing configuration uses implementation-oriented biomedical sensor modules selected according to affordability, accessibility, integration capability, and compatibility with embedded IoT operation. Their inclusion does not establish clinical-grade measurement performance.

The present study does not provide clinical validation. Local evaluation of predefined abnormal measurement conditions is an architectural monitoring function and should not be interpreted as diagnosis or replacement of qualified healthcare professionals.

Mbanza-Ngungu was used as the reference context. Applicability elsewhere should be reassessed through local requirements analysis. A production-scale implementation would also require deeper investigation of cybersecurity, privacy, authentication, authorization, device management, interoperability, maintainability, and regulatory compliance.

Future work should proceed through implementation of a more complete physical prototype, controlled technical evaluation of communication performance and synchronization behavior, extension to multiple physical nodes and realistic community network conditions, and assessment of power consumption, battery autonomy, data integrity, security, and recovery from simultaneous disruptions.

Deployment-oriented research should involve healthcare professionals and community stakeholders to evaluate usability, acceptability, workflow integration, maintainability, and practical relevance. Any progression toward clinical use would additionally require appropriate ethical approval, clinically suitable sensing equipment, comparison against reference measurements, and formal clinical validation.

8. Conclusion

This study proposed IoT-CKS, a requirements-driven Edge-Fog-Cloud architecture for infant health monitoring in resource-constrained community settings, using Mbanza-Ngungu in the Democratic Republic of the Congo as the reference context for requirements identification.

Using a Design Science Research approach, contextual constraints were progressively translated into system requirements, design principles, functional responsibilities, architectural layers, and implementation-oriented technology choices. The Edge supports physiological data acquisition, preprocessing, local evaluation of predefined abnormal measurement conditions, immediate local alerting, and temporary storage. The Fog provides intermediate coordination, messaging, aggregation, communication management, and support for deferred synchronization. The Cloud provides persistent storage, broader analysis, visualization, supervision, and administrative services.

A central characteristic of IoT-CKS is that temporary loss of Cloud connectivity is not treated as equivalent to complete loss of monitoring functionality. Essential local functions can remain available during communication disruption, while data awaiting remote transmission can be temporarily retained and synchronized after connectivity is restored.

The scientific contribution does not lie in individual Edge, Fog, Cloud, MQTT, or sensing technologies. It lies in their explicit, traceable, and context-aware integration into a resilience-oriented architecture, where functional allocation is derived from requirements associated with the intended operating environment.

The principal architectural contribution can be summarized as: Contextual constraints -> Requirements -> Design principles -> Functional allocation -> Edge-Fog-Cloud architecture -> Resilience mechanisms -> Implementation technologies.

The proposed architecture provides a foundation for subsequent implementation and experimental evaluation. IoT-CKS should be regarded as a technical and architectural DSR artefact, not as a clinically validated or certified medical device.

References

[1] Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. Computer Networks, 54(15), 2787-2805. https://doi.org/10.1016/j.comnet.2010.05.010

[2] Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7), 1645-1660. https://doi.org/10.1016/j.future.2013.01.010

[3] Islam, S. M. R., Kwak, D., Kabir, M. H., Hossain, M., & Kwak, K. S. (2015). The Internet of Things for health care: A comprehensive survey. IEEE Access, 3, 678-708. https://doi.org/10.1109/ACCESS.2015.2437951

[4] Joyia, G. J., Liaqat, R. M., Farooq, A., & Rehman, S. (2017). Internet of Medical Things (IoMT): Applications, benefits and future challenges in healthcare domain. Journal of Communications, 12(4), 240-247. https://doi.org/10.12720/jcm.12.4.240-247

[5] Dwivedi, R., Mehrotra, D., & Chandra, S. (2022). Potential of Internet of Medical Things (IoMT) applications in building a smart healthcare system: A systematic review. Journal of Oral Biology and Craniofacial Research, 12(2), 302-318. https://doi.org/10.1016/j.jobcr.2021.11.010

[6] Alharbi, H. A., Yosuf, B. A., Aldossary, M., & Almutairi, J. (2023). Energy and latency optimization in Edge-Fog-Cloud Computing for the Internet of Medical Things. Computer Systems Science and Engineering, 47(1), 1299-1319. https://doi.org/10.32604/csse.2023.039367

[7] Yıldırım, E., Cicioğlu, M., & Çalhan, A. (2023). Fog-cloud architecture-driven Internet of Medical Things framework for healthcare monitoring. Medical & Biological Engineering & Computing, 61(5), 1133-1147. https://doi.org/10.1007/s11517-023-02776-4

[8] Verma, H., Chauhan, N., & Awasthi, L. K. (2023). A comprehensive review of Internet of Healthcare Things: Networking aspects, technologies, services, applications, challenges, and security concerns. Computer Science Review, 50, 100591. https://doi.org/10.1016/j.cosrev.2023.100591

[9] Bonomi, F., Milito, R., Zhu, J., & Addepalli, S. (2012). Fog computing and its role in the Internet of Things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing (pp. 13-16). ACM. https://doi.org/10.1145/2342509.2342513

[10] Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge computing: Vision and challenges. IEEE Internet of Things Journal, 3(5), 637-646. https://doi.org/10.1109/JIOT.2016.2579198

[11] Satyanarayanan, M. (2017). The emergence of edge computing. Computer, 50(1), 30-39. https://doi.org/10.1109/MC.2017.9

[12] Khan, W. Z., Ahmed, E., Hakak, S., Yaqoob, I., & Ahmed, A. (2019). Edge computing: A survey. Future Generation Computer Systems, 97, 219-235. https://doi.org/10.1016/j.future.2019.02.050

[13] Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. MIS Quarterly, 28(1), 75-105. https://doi.org/10.2307/25148625

[14] Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. Journal of Management Information Systems, 24(3), 45-77. https://doi.org/10.2753/MIS0742-1222240302

[15] Wieringa, R. J. (2014). Design science methodology for information systems and software engineering. Springer. https://doi.org/10.1007/978-3-662-43839-8

[16] International Organization for Standardization, International Electrotechnical Commission, & Institute of Electrical and Electronics Engineers. (2018). ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering (2nd ed.).

[17] Object Management Group. (2017). OMG Unified Modeling Language (OMG UML), Version 2.5.1. OMG Document Number formal/2017-12-05.

[18] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of Things: A survey on enabling technologies, protocols, and applications. IEEE Communications Surveys & Tutorials, 17(4), 2347-2376. https://doi.org/10.1109/COMST.2015.2444095

[19] OASIS. (2019). MQTT Version 5.0. OASIS Standard.