Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 3278371, 17 pages
https://doi.org/10.1155/2022/3278371

Research Article

A Suite of Design Quality Metrics for Internet of Things by Modelling Its Ecosystem as a Schema Graph

Zeest Waris,[1] Abdul Jaleel,[2] Muhammad Shoaib,[1] Natasha Nigar,[2] and Douhadji Abalo[3]

[1]*Department of Computer Science, University of Engineering and Technology, Lahore 54890, Pakistan*
[2]*Department of Computer Science (RCET Campus, GRW), University of Engineering and Technology, Lahore 52250, Pakistan*
[3]*University of Lomé, P.O.Box 1515, Lomé, Togo*

Correspondence should be addressed to Douhadji Abalo; douhadjiabalo@gmail.com

Received 4 March 2022; Accepted 22 April 2022; Published 7 June 2022

Academic Editor: Muhammad Faisal Nadeem

Internet of Things (IoT) infrastructure connects consumer electronics, household appliances, and other smart gadgets. For designing IoT systems and applications, different architectures and protocols are being used. The design quality of an IoT system and its services is assessed with quality of service (QoS) factors such as complexity, functional appropriateness, performance, efficiency, compatibility, maintainability, portability, and usability. The existing methods in the literature focus on measuring the quality of an IoT system during execution time; however, there is a need to define some standards that may be used to determine the quality of an IoT system from its design. This work models the IoT EcoSystem as a schema graph and presents a suite of metrics to determine the design quality of an IoT system and its services under the defined QoS factors. The presented metrics are mapped to the ISO-9126 QoS factors to ensure the standards and quality of IoT services during the design phase. The proposed metrics are validated with benchmarking on a case study smart healthcare IoT system. The results proved that the presented metrics are practical to quantify the quality of IoT design models under QoS factors, and the suite of metrics is able to compare differently designed IoT services.

1. Introduction

For computing applications, design decisions have a substantial impact on the final product's quality [1]. The Internet of things (IoT) domain has distinct design requirements because of the existence of many communication protocols, terminologies, and architectures and involves factors like heterogeneity, diverse application domains, varied functionalities, and components [2]. During the design process of an IoT application or system, quality criteria and related measurements must be observed. Determining the quality of IoT applications and services is considerably different from conventional software systems [3]. Various IoT architectures in the literature lead to challenges when designing IoT systems and applications. The resource-constrained IoT devices, being arrayed in vastly dynamic settings and union of a range of technologies, cause conventional design standards and models to be insufficient to measure the quality of service (QoS) in IoT applications [3–6].

Graph-based metrics are an important tool to quantify software design models' quality. The IoT EcoSystem is a design model that interconnects the heterogeneous smart devices in a broad network of distinct components to operate efficiently [7]. The IoT EcoSystem interoperates web-enabled smart devices to collect, retrieve, send, and act on data from their surroundings by using the embedded system processors, sensors, and communication hardware [8, 9]. The scope of this work is to model the IoT EcoSystem as a schema graph to define the quality measurement metrics for the IoT domain such that the service quality of an IoT system or application may be evaluated from its design model. A research matrix for the analysis of existing related research works is given in Table 1. It summarizes the current

Table 1: Research matrix for the analysis of existing relevant research works.

Author, year	Research area	Quality factors	Design phase metric
Li et al. [10] (2014)	Service-oriented IoT model for QoS scheduling	Performance and cost	3-layer architecture for QoS
Al-Fuqaha et al. [11] (2015)	Identified IoT quality factors	Availability, reliability, mobility, security, privacy, interoperability, confidentiality, scalability, fault tolerance, and operations management	Not covered
Kiruthika and Khaddaj [12] (2015)	Designing quality models for IoT systems	Security, performance, usability, reliability, robustness, interoperability, and scalability	Not covered
Kim [13] (2016)	Quality model for IoT application	Functionality, reliability, efficiency, and portability	Not evaluated
Costa et al. [5] (2016)	Framework to model IoT application for QoS	Performance, reliability, and availability	Design and analysis process
Tambotoh et al. [14] (2016)	Governance IoT framework	Functional suitability, reliability, compatibility, security, usability, maintainability, performance efficiency, and portability	Software quality model for loT
White et al. [4] (2017)	Systematic mapping of QoS factors for IoT	Efficiency, performance, compatibility, functional suitability, reliability, usability, security, portability, and maintainability	Not covered
Tanganelli et al. [15] (2018)	Analysis to enforce QoS	Resiliency, reliability, and latency	Not covered
Singh and Baranwal [16] (2018)	QoS metrics based on computing, communication, and things as three IoT pillars	Reliability, interoperability, flexibility, availability, accuracy, stability, response time, sensitivity, precision, and scalability	Not covered
Bures et al. [17] (2018)	QoS issues and challenges in IoT environment	Privacy, security, reliability, interoperability, and integration	Not covered
Zavala et al. [18] (2019)	IoT autonomic model	Device's heterogeneity, scalability, and ubiquity	Not covered
Suryanegara et al. [19] (2019)	Framework for measuring quality of experience in IoT	Functional suitability, consistency, convenience, efficiency, and integration	Not covered
Jaleel et al. [20] (2020)	Autonomic interoperability manager for IoT	Interoperability	Not covered
Samann et al. [21] (2021)	Fog and cloud-based IoT computing with QoS provision	Latency, reliability, throughput, and network usage	Not covered
This work (2022)	Design quality metrics for IoT	Complexity, functional suitability, performance, efficiency, compatibility, maintainability, portability, and usability	Metrics are defined, evaluated, and mapped to QoS factors

works on the base of (i) the IoT research area covered by the given works, (ii) the quality factors suggested for IoT applications, and (iii) whether the authors cover the design phase metrics. The research matrix shows that very few works have focused on defining and evaluating the quality of IoT services at the design phase through metrics. However, most works elaborate on the quality factors to standardize the IoT architectures and models.

The literature survey categorized the related works into four levels based on the qualitative and quantitative measurements criteria presented in specific research work. At the top, the services inside an IoT infrastructure are divided into information aggregation services, identity-related services, ubiquitous services, and collaborative-aware services [2]. The IoT services are usually composable and reconfigurable, leading to a dynamic environment requiring explicit support at different levels to ensure high quality for its users [15]. At the second level, Tambotoh et al. [14] identified the IoT characteristics as heterogeneity, resource-constrained devices, collaboration with hardware and resources, network mobility, embedded and adaptive devices, and design to monitoring IoT devices. At the next level, QoS factors for IoT systems and services are presented [3, 4, 6, 17, 22] based on ISO-9126 model [23]. It includes complexity, functional suitability, performance, efficiency, compatibility, maintainability, portability, and usability. At the fourth level, there are some quality analysis related works, including a 3-layer IoT architecture for QoS [10], a design and analysis process [5], and a software quality model [14].

It is evident from the data in Table 1 that the existing research focused on identifying the quality factors for IoT, whereas literature lacks quantitative metrics for design quality evaluation of IoT models and services. Existing research has mapped different features of IoT to the QoS parameters; however, there is a lack of design quality measuring metrics required for estimating the services quality of IoT systems at an early stage. This research aims to propose a suite of design quality metrics to ensure service quality in the IoT. More specifically, we have asked the following research questions: (a) What are the potential relationships between IoT service categories and identified IoT characteristics; (b) What are the QoS factors that affect the QoS in a heterogeneous IoT infrastructure; (c) How to define design metrics for IoT applications, needed to ensure the quality of services; (d) What are the possible metrics

mappings to the IoT quality characteristics and services; (e) How should the presented design quality metrics be evaluated; (f) What is the advantage of evaluating an IoT system under presented design quality metrics.

This research modeled the IoT Ecosystem as a schema graph and defined a suite of design metrics to ensure the quality of IoT services. The newly defined metrics for IoT design quality assessments are IoT transactional complexity, IoT design complexity, IoT transactional network load, IoT interoperability resolving network complexity, IoT interoperability resolving network load, and IoT reusability factor. As given in Table 1, this work presented and evaluated the design quality metrics and mapped them to the QoS parameters. In a summarized form, we make the following contributions to the domain of IoT for design quality evaluation:

(i) Design quality metrics are defined for the IoT domain to develop an early stage (design phase) acceptance criteria for quality assurance of services provided by IoT systems.

(ii) The presented metrics are mapped to the ISO-9126 QoS factors and, in return, mapped to the identified characteristics of IoT systems which are then mapped to the service categories in IoT systems.

(iii) We validate the proposed metrics with a smart healthcare IoT system as a case study. The results proved that the presented metrics are practical to quantify the quality of IoT design models under QoS factors.

(iv) The metrics are viable to determine the design quality of services in the IoT applications. Also, these metrics are useful in comparing various designs of IoT services in an IoT infrastructure for better quality.

The rest of this article is organized as follows. The next section is dedicated to the literature review. Section 3 defines the Internet of Things EcoSystem as an abstract graph and introduces its schema representation. The design quality metrics for the quality services in the Internet of things are proposed in section 4. The evaluation and results are given in section 5. Finally, section 6 concludes the article.

2. Literature Review

This section reviewed existing works related to IoT architecture(s), models, and those covering parameters and attributes for QoS in IoT. For IoT architecture, De et al. [24] proposed a dynamic creation of services and their testing in the IoT-A reference architecture-based IoT environment. The presented architecture is semantic oriented and test-driven and provides self-managing and testing for IoT services and creates dynamic test cases generation and execution. Yaqoob et al. [25] presented a three-layered IoT architecture with layers including application, transport, and network. They summarized various IoT architectures, including peer-to-peer, software-defined architecture, and network-based and CloudThings architecture. These architectures aim to provide QoS in heterogeneous wireless network environments and accelerate the software development process.

Li et al. [10] recognized that the varied and huge amount of devices in the IoT infrastructure makes it tough to fulfill QoS requirements. The services in IoT are reconfigurable and provide quality-aware scheduling of the service-oriented IoT architectures that increase the performance and minimize the cost. Al-Fuqaha et al. [11] highlighted the key IoT challenges and QoS criteria in terms of reliability, performance management, mobility, availability, scalability, privacy, security, interoperability, confidentiality, fault tolerance, safety, resource management, operations management, architecture standardization, protocols, and network addressing and identification.

Kiruthika and Khaddaj [12] highlighted the lack of standardization in IoT paradigm attributes like self-healing hardware, functional and nonfunctional requirement, and heterogeneous mix of devices as challenges for defining the QoS model for IoT. The identified QoS factors for IoT applications are security, performance, usability, reliability, interoperability, robustness, and scalability. The QoS features vary according to the dynamic environment. Kim [13] described that the IoT applications are a complex mixture of a variety of heterogeneous devices and technologies. They derive a practical model based on the qualities of IoT applications. The QoS factors are identified as functionality, reliability, efficiency, and portability. The metrics for these QoS factors can evaluate the quality of IoT applications.

Costa et al. [5] highlighted two major design issues of the IoT. One is the representation of complex heterogeneous entities. The other is the unavailability of the method to verify the QoS in the early design phase due to resource-constrained devices deployed in a highly dynamic and unreliable environment. Performance, reliability, and availability are the features to be considered. Tambotoh et al. [14] presented an IoT software quality model that is established on ISO/IEC 25010 and information quality characteristics of COBIT 4.1. The updated version of the ISO/IEC 9126 model is defined that determines the requirements of software product quality and evaluates them. Quality assurance is considered necessary for the security and safety of the system and users in IoT. The model evaluates the internal and external properties, and includes eight quality factors, namely, functional suitability, reliability compatibility, security, usability maintainability, performance efficiency, and portability.

Thomas and Rad [22] described that the essential quality metrics to analyze and evaluate for measuring performance in the IoT system are availability, usability, reliability, and maintainability, as mentioned. White et al. [4] used ISO/IEC 25010 and OASIS WSQM as quality models for mapping of QoS factors. The quality factors for QoS evaluation were considered based on the ISO/IEC 25010 quality model. The reliability, efficiency, functional stability, and performance are the most considered/studied quality factors; however, many factors like maintainability, security, and compatibility that are critical for the proper working of the IoT applications and systems have been neglected. Moreover, it was identified that the most addressed layers are the network layer and physical layer, whereas deployment, middleware, and cloud layers lack primary studies.

Because communication, computing, and things are the three pillars of IoT, Singh and Baranwal [16] classified QoS components into QoS of communication, QoS of things, and QoS computing. Weight, interoperability, flexibility, reliability, availability, overall accuracy, stability, response time, range, sensitivity, precision security, and other communication quality factors include bandwidth, throughput, efficiency, network connection time, monetary cost, availability, security and privacy, interoperability, service level agreement, monitoring, and reliability. Communication quality factors include weight, interoperability, flexibility, reliability, availability, overall accuracy, stability, response time, range, sensitivity, and precision security. Scalability, dynamic availability, dependability, pricing, response time, security and privacy, capacity, customer support facility, user feedback, and review are all aspects of computing QoS.

Zhohov [26] proposed a quality of experience (QoE) model for dealing with the Industrial IoT application and services. The current models for evaluating the QoE are not suitable for IIOT applications as they mainly target multimedia services. Industry-related KPIs are defined to deliver the QoE for IIoT by associating the technology and business domains. They proposed QoE layered model which forecasts efficiency, productivity, safety, and reliability as Industrial KPIs for QoS. Network performance evaluation is more complicated as each IIoT has specific QoS assurances than conventional systems. Bures et al. [17] discussed various issues and challenges faced in ensuring the QoS in the IoT environment. Many issues in quality assurance have arisen as a result of the IoT's development, including privacy and security of user information and data. Reliability, interoperability, and integration issues created a need to develop a methodology to ensure quality assurance in IoT. Systematical testing and quality assurance methods need improvement.

Zavala et al. [18] proposed an autonomic IoT infrastructure for transiting from partial-autonomic characteristics of IoT applications to complete autonomically. The inclusion of the architectural and functional blocks introduces the self-star properties. Self-configuration, self-adaptability, self-healing, self-optimization, and self-protection deal with the challenges of a dynamic, heterogeneous mix of devices, human-computer interaction, interoperability of communication protocols, scalability, and ubiquity.

Suryanegara et al. [19] established a framework for assessing IoT service QoE. The Absolute Category Rating with Hidden Reference (ACR-HR) scale served as the foundation for the framework. It assesses the quality based on the rating given by users based on their experiences. Users provide the score both before and after the implementation of the system. The user experience was evaluated via conduction of survey which is based on mean opinion score (MoS) and calculation of results is done using differential MoS based on ACR-HR. Samann et al. [21] reviewed the available techniques for QoS in IoT academia applications. Metrics considered for provisioning QoS using cloud computing in academia are latency, reliability, throughput, and network usage. However, it is not a unified process for providing all these factors.

The literature review has shown that the heterogeneity and variety of application domains for IoT have resulted in varied specifications leading to a complex IoT system with different performances. Due to this reason, the design and architecture of IoT are affected and are not standardized, thus resulting in the different architectures [27]. Due to many application domains and heterogeneous entities communicating with one another using varied protocols, there are many standards for IoT architecture that leads to the challenges in designing the IoT applications. The resource-constrained diverse mobiles and other devices resulted in the QoS concern [28]. The QoS in an IoT application is constrained by heterogeneity, resource-constrained devices, collaboration with hardware, natural human interfaces, networked mobility and volatile connectivity, embedded and adaptive devices, and design to monitor IoT devices [14]. It is necessary to specify the quality characteristics of IoT applications and evaluate them to determine whether or not they are according to the design standards.

3. Internet of Things EcoSystem as Abstract Graph

This work is based on the IoT EcoSystem concept of Bansal and Kumar [7] that puts all the heterogeneous components of IoT together to build an efficient system. For defining measurement criteria in terms of design quality metrics for the said attributes, we define the IoT EcoSystem as an abstract graph. The IoT EcoSystem concept presented by Bansal and Kumar [7] is converted into an interaction graph 'G' as given in Figure 1. The sensors and actuator devices work at the base layer of the IoT EcoSystem. The sensor devices collect information about environmental/physical parameters and pass the data to a gateway node (working at the upper layer). The actuator devices take instructions from the gateway and act on the linked entities/parameters/environmental attributes. The gateway manages the data flow between sensors/actuators devices. The sensors and actuators use different communication protocols to interact with the gateway. Hundreds to thousands of sensors and actuators can be managed via the gateway that filters and formats the data coming from/to sensors/actuators.

Above the gateway layer, a controller interacts with the middleware. The controller controls data coming to gateways from the middleware or IoT platform. The controller is responsible for the high-level processing of data. It classifies, computes, and converts data into information. A controller can manage hundreds of gateways. The IoT middleware does various activities like data analysis, saving data into the database server, preparing reports and graphs, and ensuring privacy and security. It manages and controls the IoT system. The cloud supports the data available in the IoT middleware. All applications avail the services and the analytical statistics delivered by the middleware through the application programming interface (API). The application provides the user view of the IoT environment.

3.1. Schema Representation of Internet of Things EcoSystem. For IoT EcoSystem's schema representation based on the abstract graph (Figure 1), we define the following terms for each layer of IoT EcoSystem:

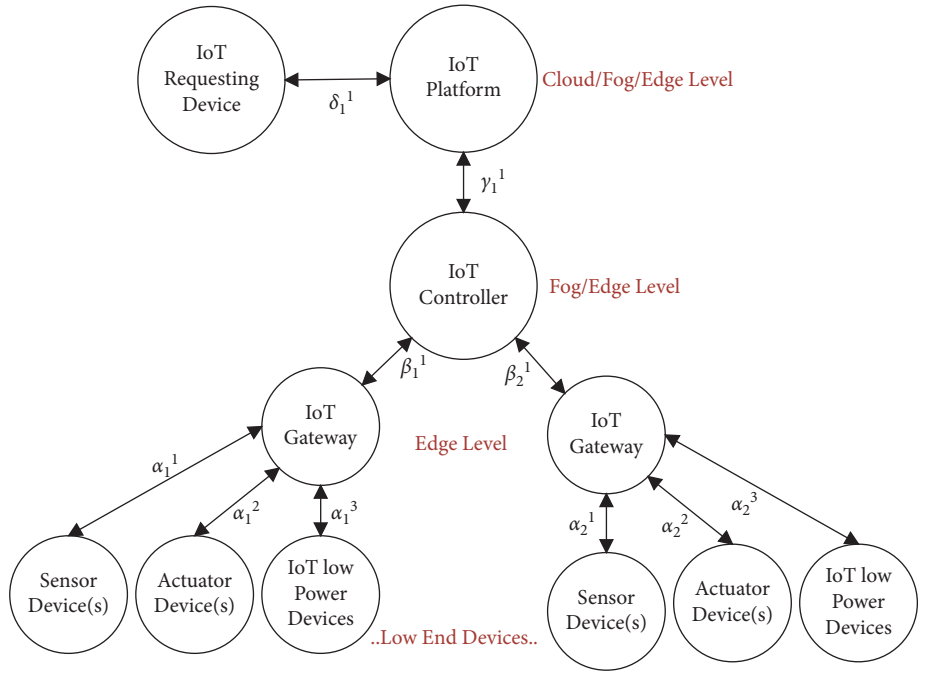

Figure 1: Abstract graph for interactions in IoT EcoSystem.

(i) D_n: LowEnd Device
G_p: Gateway Devices
C_q: Controller Devices
P_r: IoT-Platform/Middleware
U_t: User/Doctor Gadgets

The integer numbers p, q, r, t, and n denote the number of devices at the associated level of IoT EcoSystem.

To define measurements related to the design quality of the IoT services, the interaction graph 'G' in (Figure 1) is formally represented in terms of its vertices V(G), given in

$$V(G) = \{D_u\} \cup \{G_v\} \cup \{C_w\} \cup \{P_x\} \cup \{U_y\}, \quad (1)$$

where $1 \le u \le n$, $1 \le v \le p$, $0 \le w \le q$, $0 \le x \le r$, and $1 \le y \le t$, the terms u, v, w, x, y are integers to de note index-number of Low End Devices, Gateway Devices, Controller, IoT-Platform, and User Devices, respectively.

Here, an important thing to note is that more than one edge may originate from a vertex, i.e., IoT modules of the graph. So, the bidirectional edges inside the ADT Graph denote the under-given interactions between different vertices and are represented with the symbols mentioned against each edge.

Gateway Device $(G_p) \leftrightarrow$ Low-End Device ——— $\alpha_v^{i_v}$
Controller Device $(C_q) \leftrightarrow$ Gateway Device ——— $\beta_w^{i_w}$
Platform Device $(P_r) \leftrightarrow$ Controller Device ——— $\gamma_x^{i_x}$
User Device $(D_n) \leftrightarrow$ Edge/Fog/Cloud ——— $\delta_y^{i_y}$

The edges' subscript integers v, w, x, and y are denoting the originating vertex number for α, β, γ, and δ edges, respectively,

where $1 \le v \le p$, $1 \le w \le q$, $1 \le x \le r$, and $1 \le y \le t$, whereas the superscript integers i_v, i_w, i_x, and i_y are denoting the edge number originating from vertices G_p, C_q, P_r, U_t, and D_n, respectively (as a counter of multiple edges from a node).

The edges are iterated in following fashion (example iterations are given for $\alpha_v^{i_v}$):

$\alpha_1^1, \alpha_1^2, \alpha_1^3, \ldots\ldots \alpha_1^{i_1}$, $\alpha_2^1, \alpha_2^2, \alpha_2^3, \ldots\ldots \alpha_2^{i_2}$, $\alpha_3^1, \alpha_3^2, \alpha_3^3, \ldots\ldots \alpha_3^{i_3}$
:: ::
$\alpha_v^1, \alpha_v^2, \alpha_v^3, \ldots\ldots \alpha_v^{i_v}$

These edge weights are measured in terms of the amount of data (in bytes) exchanged during a single interaction activation.

4. Design Quality Metrics for Quality Services in the Internet of Things

IoT applications generate a set of read-write operations representing a certain service in a transactional manner [29]. This work considers such operations as IoT Transactional Activities, where an IoT Transaction is a set of ordered pairs of request and response-based communications among IoT entities/modules. So, interactions among different components of an IoT EcoSystem are involved in a transaction. In terms of interaction edges emanating from distinct vertices of the schema graph, an IoT transaction (T_r) is described as follows:

$$T_r = \left\{\left(\left\{\alpha_i^{i'}\right\} \epsilon T_r\right) \subseteq \alpha_v^{i_v}\right\} \cup \left\{\left(\left\{\beta_j^{j'}\right\} \epsilon T_r\right) \subseteq \beta_w^{i_w}\right\} \cup \left\{\left(\left\{\gamma_k^{k'}\right\} \epsilon T_r\right) \subseteq \gamma_x^{i_x}\right\} \cup \left\{\left(\delta_l^{l'} \epsilon T_r\right) \subseteq \delta_y^{i_y}\right\}. \quad (2)$$

4.1. IoT Transactional Complexity Metric (ITCM). We define the complexity of an IoT transaction as the number of its constituting interactions that originates from various vertices (modules of the IoT system, including IoT devices and network nodes). IoT transactional complexity metric (ITCM), defined in (3), counts the number of interactions involved in the transaction by taking the edge weight for each of the interactions as a unit value.

$$\begin{aligned} ITCM = \sum\left(\left\{\alpha_i^{i'}\right\} \in T_r\right) + \sum\left(\left\{\beta_j^{j'}\right\} \in T_r\right) \\ + \sum\left(\left\{\gamma_k^{k'}\right\} \in T_r\right)) + \sum\left(\left\{\delta_l^{l'}\right\} \in T_r\right) \\ \text{where } \alpha_i^{i'} = \beta_j^{j'} = \gamma_k^{k'} = \delta_l^{l'} = 1. \end{aligned} \tag{3}$$

ITCM metric is used to measure the complexity of an IoT transaction. For determining the Total Transactional Complexity of an IoT system, the metric defined in (4) simply adds the complexity of each transaction available in an IoT system.

$$\text{Total Transactional Complexity} = \sum_{i=1}^{n} ITCM_i. \tag{4}$$

The average transactional complexity is then measured by

$$\text{Average Transactional Complexity} = \frac{\sum_{i=1}^{n} ITCM_i}{n}. \tag{5}$$

A rise in the value of these measures indicates an increase in complexity, which impacts understandability and maintainability. The average transactional complexity scale is based on the works [30, 31] on software complexity. The following are the ranges on the average transactional Complexity scale:

$$\text{Low}\,(<5), \text{Medium}\,(5 \text{ to } 10), \text{High}\,(>10). \tag{6}$$

In general, the longer an IoT transaction is, the longer it takes to complete it via the network. As a result, latency will be noticed. Each additional interaction wastes time and causes delays. By integrating numerous unnecessary encounters into a single interaction, the Transactional Complexity score can be decreased. The measurements assist in selecting a better alternative for an IoT transaction among the many accessible options.

4.2. IoT Design Complexity Metric (IDCM). On the base of interactions among different components of an IoT EcoSystem, we define the IoT Design Complexity Metric given as follows:

$$\text{IDCM} = \sum_{v=1,i_v=1}^{p,p_v} \alpha_v^{i_v} + \sum_{w=1,i_w=1}^{q,q_w} \beta_w^{i_w} + \sum_{x=1,i_x=1}^{r,r_x} \gamma_x^{i_x} + \sum_{y=1,i_y=1}^{t,t_y} \delta_y^{i_y}. \tag{7}$$

Here, the edge weight value for each interaction is taken as a unit value. The average design complexity for an IoT system is then determined as given in

$$\text{Average Design Complexity} = \frac{\text{Total number of Interactions}}{\text{Total number of Devices}}. \tag{8}$$

This metric determines how complex the IoT system will be, with an increasing number of IoT devices and their interactions. The complexity scale defined for the transactional complexity metric applies here as well.

4.3. IoT Transactional Network Load (ITNL). An IoT transaction involves interactions among different components of an IoT EcoSystem involved in a transaction. IoT transactional network load (ITNL) is proportional to the complexity of an IoT transaction, determined by (7) by taking the edge weight for each interaction as a unit value. The ITNL value for an IoT transaction is determined with (3) by taking the actual edge weight values. The transaction's edge weights add to calculate the ITNL metric, given as

$$\begin{aligned} ITNL = \sum\left(\left\{\alpha_i^{i'}\right\} \in T_r\right) + \sum\left(\left\{\beta_j^{j'}\right\} \in T_r\right) \\ + \sum\left(\left\{\gamma_k^{k'}\right\} \in T_r\right) + \sum\left(\left\{\delta_l^{l'}\right\} \in T_r\right). \end{aligned} \tag{9}$$

For determining the total transactional network load of an IoT system, the metric defined in (10) simply adds the network load of each transaction happening in an IoT system.

$$\text{Total Transactional Network Load} = \sum_{i=1}^{n} ITNL_i. \tag{10}$$

An average Transactional Network Load is determined by the following formula:

$$\text{Average Transactional Network Load} = \frac{\sum_{i=1}^{n} ITNL_i}{n}. \tag{11}$$

More value of the transactional network load for an IoT service means overburdening the IoT network and may choke the system. Transactional network load value may be reduced by reducing the data size in request and response if possible. The metrics help choose a better IoT transaction from the available option, which brings the least data burden for the IoT network. The ITNL value at each hope must be within the available bandwidth.

4.4. IoT Communication Channel Demand (ICCD). IoT communication channel demand for a network hop is the byte size of data traveled between its nodes. The communication channel demand is the sum of data weights of all edges activated for an IoT transaction. However, all transactions need to be considered to determine the channel demand for each hope of the IoT system. If multiple

transaction activities occur simultaneously through one hop, the channel demand depends upon the transaction with the largest data size (weight) for that edge. The IoT communication channel demand (ICCD) of an IoT design model can be determined from the schema design presented in Figure 1. It is identifiable as the maximum interaction weight between any two graph vertices (i.e., IoT modules). For determining the communication channel demand (CCD) among the low-end devices and the gateway devices, the edge weights $\alpha_v^{i_v}$ are to be considered. The metric in equation (11) chooses maximum interaction weight value for $\alpha_v^{i_v}$.

$$CCD_{\text{alpha}} = \max\left(\max\left(\beta_w^{i_w}\right), \max\left(\text{ReqSize}, \text{RespSize}\right)\right), \tag{12}$$

where, β_w^{iw} represents the packet the packet size being sent to or recieved from the upper layer and the second paramaters is the (Request/Response) packet size of the low end devices (sensor/actuator).

The communication channel demand among the gateway devices and the controller devices $(\beta_w^{i_w})$ is determined by the weights of the upper and lower links. It is measured as the maximum interaction weight value from the accumulated interaction weights $\alpha_v^{i_v}$ and the maximum of interaction weights $\gamma_x^{i_x}$, as defined in

$$CCD_{\text{beta}} = \left\{\max\left(\sum \alpha_v^{i_v}\right), \max\left(\gamma_x^{i_x}\right)\right\}. \tag{13}$$

To find the communication channel demand among the controller devices and the middleware devices $(\gamma_x^{i_x})$, the maximum interaction weight value from the interaction weights $\beta_w^{i_w}$ and the maximum interaction weight value from the interaction weights $\delta_y^{i_y}$ are to be selected. The metric is defined in

$$CCD_{\text{gamma}} = \max\left(\max\left(\beta_i^j\right), \max\left(\delta_i^j\right),\right). \tag{14}$$

The metric defined in (15) finds the communication channel demand among the platform and the user devices, i.e., $\delta_y^{i_y}$. From the interaction weights, the metric selects the maximum interaction weight value $\gamma_x^{i_x}$ (as response size) and request packet size is equal to request/response packet length of the user end device.

$$CCD_{\text{delta}} = \max\left(\max\left(\text{ReqSize}, \text{RespSize}\right), \max\left(\gamma_i^j\right)\right). \tag{15}$$

Finally, the ICCD metric selects the maximum interaction weight value from four sets of interaction weights (i.e., $\alpha_v^{i_v}$, $\beta_w^{i_w}$, $\gamma_x^{i_x}$, and $\delta_y^{i_y}$). The maximum interaction weight value amongst these interaction weights gives the maximum data size that may transfer over an interaction. Its value can be determined by the metric defined in the following equation:

$$ICCD_{IoT} = \left\{\sum \alpha_v^{i_v}, \max\left(\beta_w^{i_w}\right), \max\left(\gamma_x^{i_x}\right), \max\left(\delta_y^{i_y}\right)\right\}. \tag{16}$$

In both local and distributed IoT systems, ICCD calculations aid in determining the required bandwidth. A higher ICCD value indicates a higher cost of bandwidth acquisition. This statistic aids in determining whether or not we can pay the costs of a projected IoT system and whether or not its development is realistic. The ICCD calculation can also be used to discover any additional data being sent over the expensive lines and, if possible, minimize it. ICCD calculation also can be used to identify any extra information being sent over the costly links and can be reduced if possible. Miscalculations about the communication channel demand or guessing its demand without proper measurements may affect the efficiency of the IoT system (if a guess is lower than the required demands) or results in extra payments (if the guess is too above the required demands).

4.5. IoT Interoperability Resolving Network Complexity (IIRNC) and Interoperability Resolving Network Load (IIRNL). For interoperability of heterogeneous IoT devices, translation of data from the source device into the sink device format is performed either at the source, the sink, or some middleware device (being at the edge, fog, or cloud levels) [20]. Other interoperability operations may require protocol translation, or semantic translations, that are performed either at the source device or by a third-party device available at edge, fog, or cloud level [32]. The edges traversed during an interoperability translation process (ITP) present the interactions involved among the IoT devices. More number of traversed edges means more interoperability resolving complexity (IRC) for the ITP. By considering the edge weights as unit values, the IIRNC metric is defined as follows:

$$IRC_{ITP} = \sum\left\{\forall\alpha_p^{p'} \in ITP, \forall\beta_q^{q'} \in ITP, \forall\gamma_r^{r'} \in ITP, \forall\delta_t^{t'} \in ITP\right\} \quad \text{where } \alpha_i^{i'} = \beta_j^{j'} = \gamma_k^{k'} = \delta_l^{l'} = 1. \tag{17}$$

The IoT interoperability resolving network complexity (IIRNC) is then calculated as the sum of IRC's for all ITPs, given in the following:

$$IIRNC_{IoT} = \sum_{y=1}^{N} IRC_y. \tag{18}$$

The size of data (in bytes) sent over a one-time interaction activation is used to calculate the edge weights. The IIRNL metric is defined in equation (18), which calculates the network traffic generated during interoperability resolving action.

$$IIRNL = \sum\left\{\forall\alpha_p^{p'} \in ITP, \forall\beta_q^{q'} \in ITP, \forall\gamma_r^{r'} \in ITP, \forall\delta_t^{t'} \in ITP\right\}. \tag{19}$$

The IIRNL measure calculates the system's burden in message size generated for each interoperability operation. The worst case is when all interoperability operations activates concurrently, calculated by

$$IIRNL_{IoT} = \sum_{y=1}^{N} IRNL_y. \tag{20}$$

The interoperability performance of an IoT system is inversely proportional to IIRNL as given in the following:

$$\text{interoperability performance} \propto \frac{1}{IIRNL_{IoT}}. \quad (21)$$

If the network data traffic load exceeds the available bandwidth, the system will run slowly or perhaps become stopped. This problem can be avoided by checking for any excess information supplied in a message; otherwise, network bandwidth will need to be raised to keep the system operational.

4.6. Reusability Factor of IoT Metric. The reusability of an IoT system is based on the reuse factor of the sensors, actuators, and other low-end, middle-level, and high-level devices in different IoT transactions. The reusability metrics for each of these modules are defined as follows:

Reuse factor of sensors-RF(S): it is the number of IoT Transactions involving a specific sensor

Reuse factor of actuators-RF(A): it is the number of IoT Transactions involving a specific actuator

Reuse factor of Low-End Devices-RF (LED): it is the number of IoT Transactions in which a Low-End Device is involved

Reuse factor of Middle Devices-RF (MD): it is the number of IoT Transactions in which a Middle Device is involved

Reuse factor of High-End Devices-RF(HED): it is the number of IoT Transactions in which a High-End Device is involved

The reusability of a transaction is then measured as given in

$$\begin{aligned}\text{Transaction}_{\text{Reusability}} = {} & RF(S) + RF(A) + RF(LED) \\ & + RF(MD) + RF(HED).\end{aligned} \quad (22)$$

The total reusability factor of an IoT system is then determined from the following:

$$\text{Total Reusability}_{IoT} = \sum_{i=1}^{p} \text{Transaction}(i)_{\text{Reusability}}. \quad (23)$$

4.7. Mappings of IoT Services, Characteristics, QoS Factors, and Design Quality Metrics. In this section, we mapped the newly defined design quality metrics to the QoS factors taken from the ISO 9126 model [23] (defined for the quality assessment of computing systems and applications). The QoS factors selected for this purpose, from the works of [3, 4, 6, 17, 22], include complexity, functional suitability, performance, efficiency, compatibility, maintainability, portability, and usability. The metrics defined for design quality evaluation of IoT services are IoT design complexity metric, IoT transactional complexity metric, IoT transactional network load metric, IoT communication channel demand metric, IoT interoperability resolving network complexity metric, and IoT interoperability resolving network load metric.

As presented in Figure 2, the QoS factor complexity is quantified with design complexity, transactional complexity, and interoperability resolving network complexity. The QoS factor stability is measured with transactional complexity. For determining the efficiency and performance factor of QoS, the metrics used are transactional complexity, transactional network load, and interoperability resolving network load. Next, we quantified the compatibility QoS factor with interoperability metrics resolving network complexity and interoperability resolving network load. At the same time, the QoS factors of maintainability and portability are mapped with interoperability resolving network load. Lastly, the QoS factor of usability is determined with the metric named as reusability factor.

The mapping between the second and third layer of Figure 2 links the QoS factors with IoT characteristics. The IoT characteristics as presented by Tambotoh et al. [14] are heterogeneity, resource-constrained devices, collaboration with hardware and resources, network mobility, embedded and adaptive devices, and design to monitoring IoT devices. The IoT characteristic 'heterogeneity' is mapped with complexity, maintainability, and portability of QoS factors. The IoT 'resource-constrained devices' characteristic is mapped with QoS factors complexity, efficiency and performance, and usability. The IoT 'collaboration with hardware and resources' characteristic is linked with QoS factors stability, compatibility, and usability. IoT's 'network mobility' characteristic is attached to compatibility, maintainability, and portability. IoT's 'embedded and adaptive devices' characteristic is linked to compatibility, maintainability, portability, and usability. Lastly, the IoT characteristic of 'design to monitor IoT devices' is mapped to maintainability, portability, and usability.

The services inside an IoT infrastructure are defined by Gigli et al. [2] as information aggregation services, identity-related services, ubiquitous services, and collaborative-aware services. Although all of these services are somehow related to each of the IoT characteristics, as presented in Figure 2, we categorically mapped each IoT service to the maximally related IoT characteristic(s). Therefore, in the pattern given in Figure 2, we can map the IoT services to IoT characteristics to IoT QoS factors which are mapped to the design quality metrics presented in this work. It allows us to determine the IoT services quality from defined design quality metrics.

5. Results and Discussion

A smart city healthcare setup is taken as a case study to assess IoT service quality measuring metrics defined in this work. In our smart city healthcare model, various medical IoT devices communicate with each other and share patients' records and live data through the cloud, fog, and edge-computing paradigms. In the smart healthcare setup, a medical application connects a healthcare provider to his patient's smart medical devices, as illustrated in Figure 3. We consider a hospital ICU having a specific number of patient beds. On each patient's bed, various low-end devices as sensors are deployed to monitor the patient. The sensors include glucose level monitor (GLM), heartbeat rate (HBR)

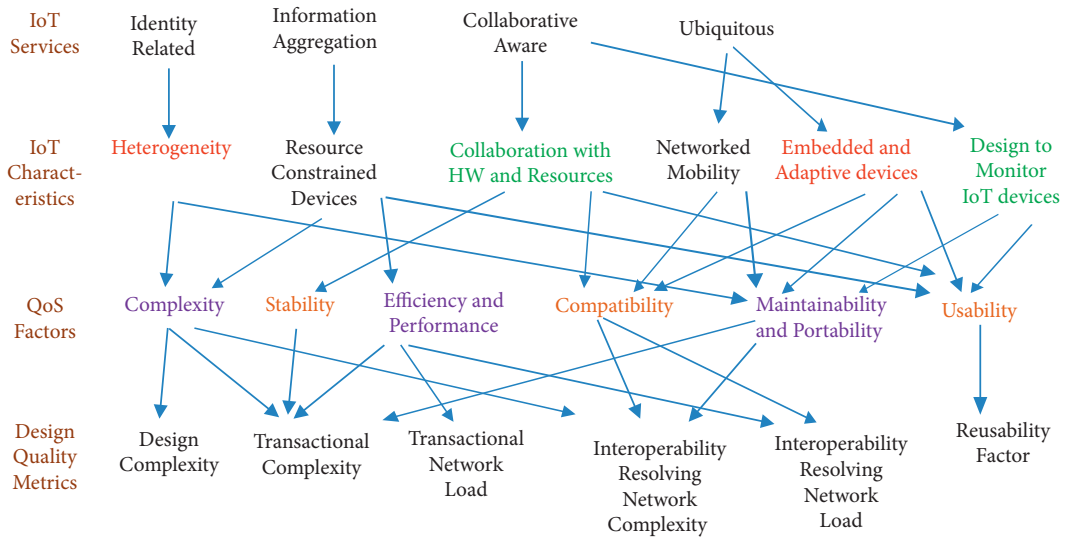

Figure 2: IoT Services [2], IoT characteristics [14], QoS factors [3, 4, 6, 22], and design quality metrics (this work).

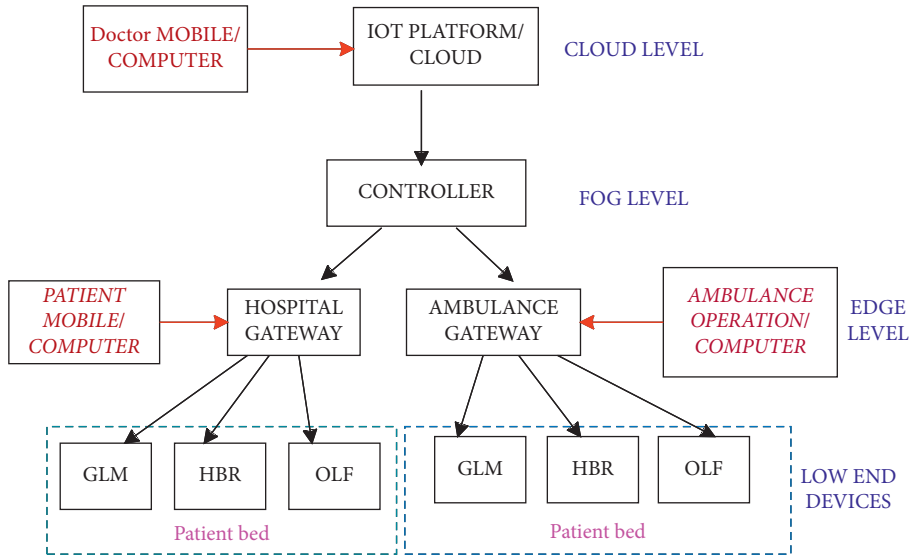

Figure 3: A healthcare IoT-enabled smart city, where a patient (in a smart intensive care unit) is wearing IoT-enabled medical devices, managed with his smartphone. The data are shared in the healthcare system through edge, fog, and cloud [20].

monitor, oxygen level flow (OLF), and blood pressure monitor (BPM). These devices are connected with the edge, fog, and cloud networks to make them accessible anywhere in the hospital, labs, and remotely to the doctor/specialist. The doctor can observe the patient through these medical devices and can give a prescription. Moreover, a specialist can set the actuators' dosage and levels remotely for the advised medicine dosage for the patient.

For evaluation, we considered different scenarios of the case study system to evaluate the metrics defined in this work. The sensors and actuator devices considered in the case study are given in Table 2. The sensors are deployed for monitoring patients' vital signs and are connected to the network for the edge, fog, and cloud connectivity. The doctor observes a patient through the sensor devices and sets the actuators to the appropriate level for administering medicine to the patient.

5.1. IoT Transactional Complexity Metric (ITCM). The IoT transactional complexity metric is evaluated with three transactional scenarios of a smart healthcare system. Scenario (i) is presented in Figure 4 where a doctor retrieves the data from only one sensor 'BPM' attached to bed 1 of hospital 1. The IoT transaction is shown in blue lines for this situation, and these data read complexity for the transaction calculated using equation (3) as four units, i.e., the number of edges involved in the transactional activity.

Table 2: Low-end sensor and actuator medical devices.

Sensors	Actuators
Glucose level monitor (GLM)	Smart infusion dosing (SID)
Blood pressure monitor (BPM)	Smart ventilator (SV)
Heart beat rate monitor (HBRM)	Automated insulin delivery (AID)
Respiration monitoring sensor (RMS)	Robotic surgery (RS)
Electro cardio graph (ECG)	Smart dental chair (SDC)
Pulse oximeter monitor (POM)	Smart hospital bed (SHB)
Temperature monitoring sensor (TMS)	Smart hospital lifts (SHLs)
Ingestible sensors (ISs)	Laser positioning equipment (LPE)
Capnography monitor sensor (CMS)	
Connected smart inhalers (CSIs)	
Fall detection sensor (FDS)	

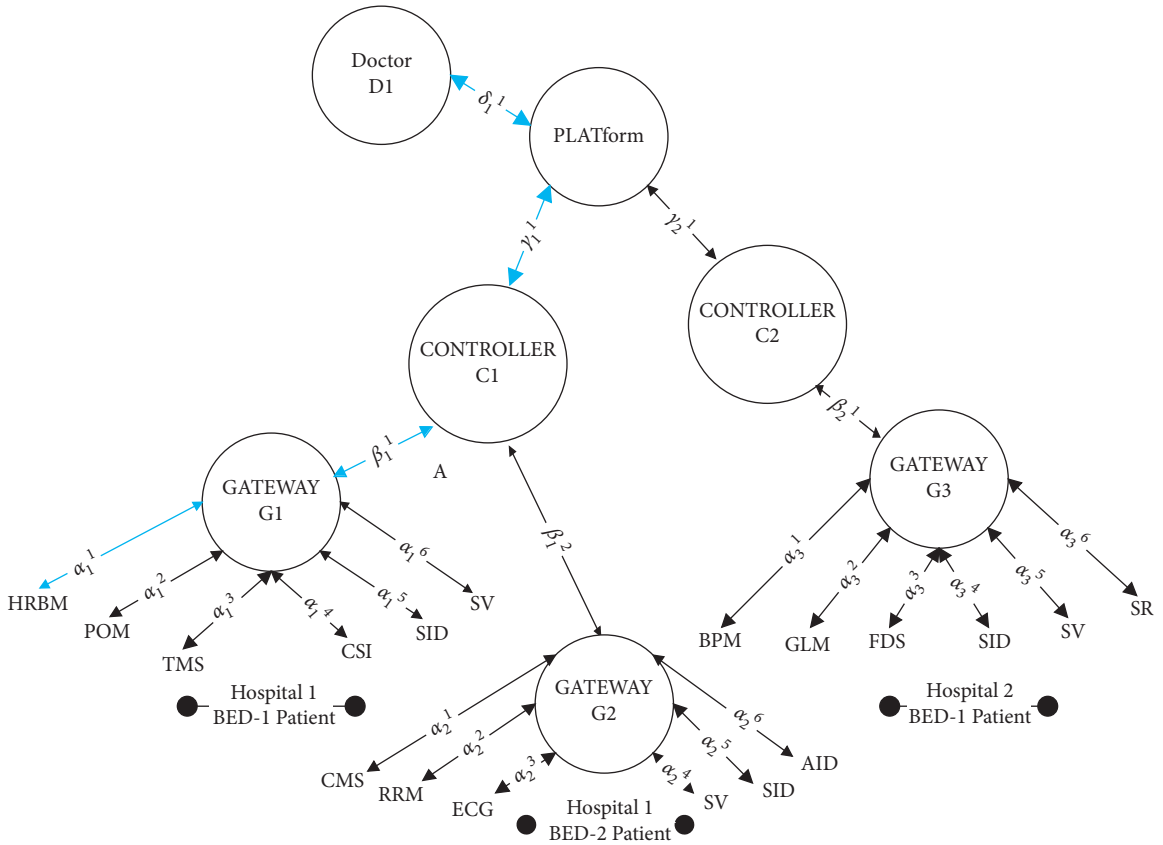

Figure 4: Scenario (i) doctor is accessing the data of only one sensor. The patient in a smart intensive care unit is on smart bed having IoT-enabled medical devices, and the doctor is monitoring and treating the patient.

$$\begin{aligned} ITCM &= \alpha_1^1 + \beta_1^1 + \gamma_1^1 + \delta_1^1 \\ &= 1 + 1 + 1 + 1 \\ &= 4 \text{ units.} \end{aligned} \tag{24}$$

The transactional response load for the sensor is also 4 units. So, the total response-request transaction complexity is 8 units.

Scenario (ii) is depicted in Figure 5 in purple edges where the doctor requests data from all sensors attached to the patient bed 1 in hospital 1. The transactional request complexity for all sensors accessed through the hospital gateway is 9 units, and the same is the transactional response complexity, i.e., 9 units.

$$\begin{aligned} ITCM &= \alpha_1^1 + \alpha_1^2 + \alpha_1^3 + \alpha_1^4 + \alpha_1^5 + \alpha_1^6 + \beta_1^1 + \gamma_1^1 + \delta_1^1 \\ &= 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 \\ &= 9 \text{ units.} \end{aligned} \tag{25}$$

The total response-request transactional complexity is 18 units.

Scenario (iii) is depicted in Figure 6 in green edges where data are retrieved from ICUs of a hospital attached via two different gateways. The transactional request complexity for

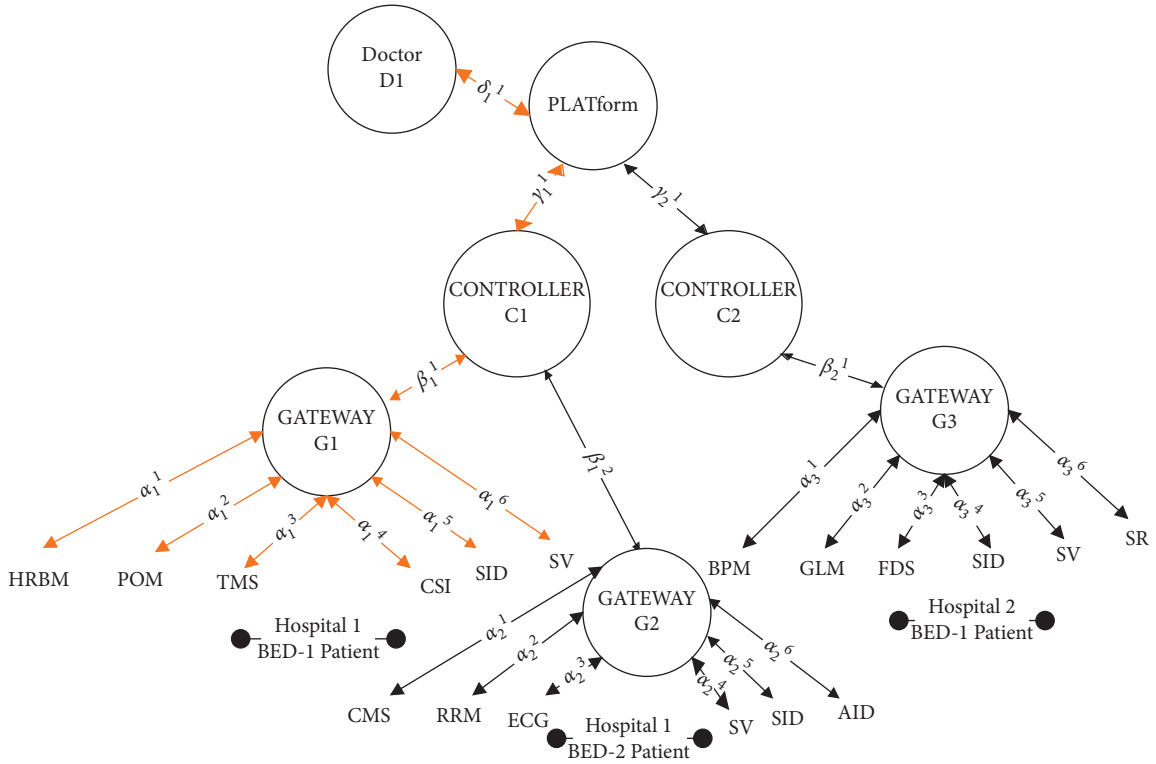

Figure 5: Scenario (ii) shown in orange edges where the doctor requests data from all sensors attached to the patient bed.

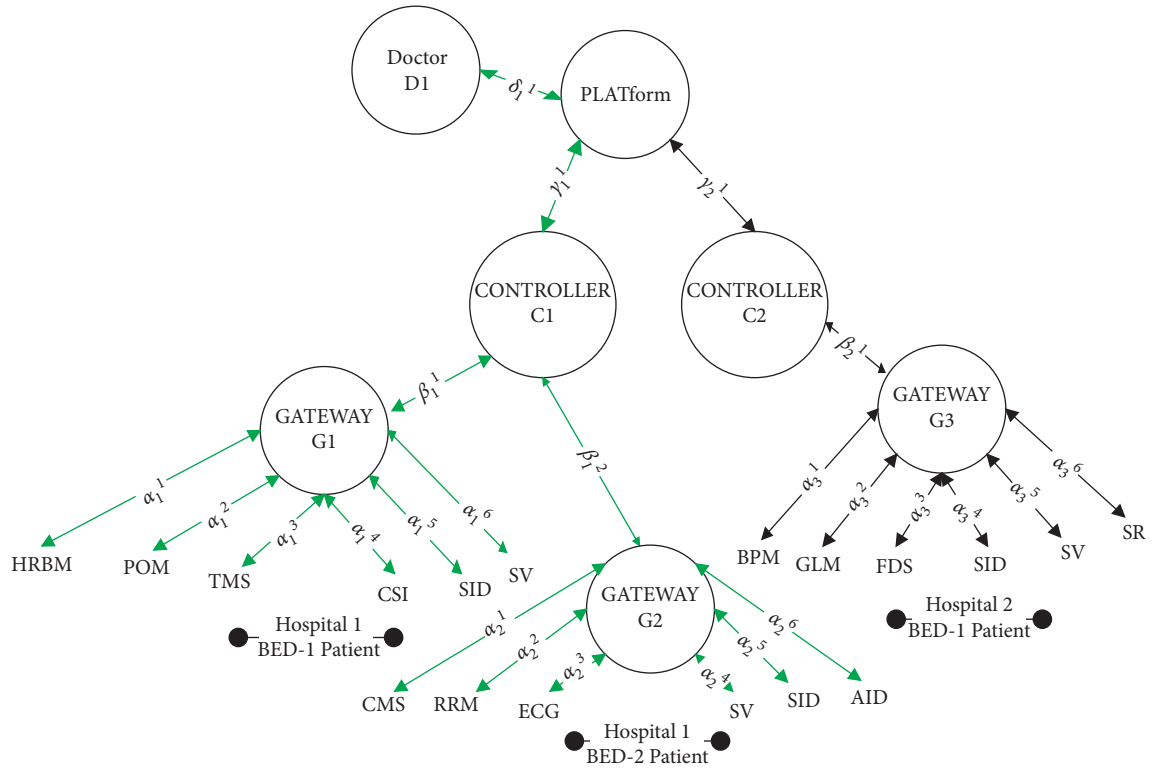

Figure 6: Scenario (iii) shown in green edges where data are retrieved from ICUs of a hospital attached via two different gateways.

all sensors accessed through the hospital gateway is 16 units, and the same is the transactional response complexity, i.e., 16 units. The total response-request transactional complexity is 32 units.

$$\begin{aligned} ITCM &= \alpha_1^1 + \alpha_1^2 + \alpha_1^3 + \alpha_1^4 + \alpha_1^5 + \alpha_1^6 + \alpha_2^1 + \alpha_2^2 + \alpha_2^3 + \alpha_2^4 + \alpha_2^5 + \alpha_2^6 \\ &\quad + \beta_1^1 + \beta_1^2 + \gamma_1^1 + \delta_1^1 \\ &= 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 \\ &= 16\,(\text{No. of all edges}). \end{aligned} \tag{26}$$

The average transactional complexity (5) for this case study model is calculated to be $(4 + 9 + 16)/3 = (29/3) = 9$, which means it is within acceptable threshold.

5.2. IoT Design Complexity Metric (IDCM) Evaluation. The design complexity metric determines the overall complexity of an IoT system from its design model. Figure 4, Figure 5, and Figure 6 depict an IoT system design model with different remote data accessing scenarios for a doctor to connect with sensors through the cloud infrastructure for accessing the patient's vital information. The design complexity of this IoT system is measured with the IDMC metric defined in (7) by taking the unit cost of the weight of each edge.

$$\begin{aligned} IDCM &= \alpha_2^1 + \alpha_2^2 + \alpha_2^3 + \alpha_2^4 + \alpha_2^5 + \alpha_2^6 + \alpha_3^1 + \alpha_3^2 + \alpha_3^3 + \alpha_3^4 + \alpha_3^5 + \alpha_3^6 \\ &\quad + \alpha_5^1 + \alpha_5^2 + \alpha_5^3 + \alpha_5^4 + \alpha_5^5 + \alpha_5^6 \\ &\quad + \beta_1^2 + \beta_1^3 + \beta_2^2 \\ &\quad + \gamma_1^1 + \gamma_2^1 + \delta_1^1 \\ &= 24\,(\text{No.of all edges}). \end{aligned} \tag{27}$$

The average design complexity (8) for this case study model is calculated to be = (24/25), which means it is less than one, which is a perfect case. So, the design model is a very simple system.

5.3. IoT Transactional Network Load (ITNL). The metric defined in (9) determines the IoT transactional network load by taking the edge weight values in place of unit costs. The data packet size for each relevant IoT device (IoT node) is considered edge weight to determine a transaction's network load. IoT transactional network load is then calculated by summing up the data loads introduced from each node involved in an IoT transaction. Table 3 presents the data size returned by each sensor/actuators device used in the case study.

The data loads (edge weights) for scenario (i) in blue edges, scenario (ii) in blue, orange edges, and scenario (iii) in blue, orange, and green edges are given in Figure 7.

The transactional network loads for the scenarios discussed under subsection 5.1 are calculated using (9) as follows:

(i) Scenario (i): ITNL = 12 bytes

(ii) Scenario (ii): ITNL = 12 + 12+6 + 16+20 + 24 = 90 bytes

(iii) Scenario (iii): ITNL = 12 + 12+6 + 16+20 + 24+8 + 7+10 + 24+20 + 16 = 1755 bytes

Using (10) and (11), the average transactional network load is 619 bytes.

5.4. IoT Communication Channel Demand Evaluation. For IoT communication channel demand evaluation, we consider the IoT design model of Figure 8, where doctors D1, D2, and D3 can access the patients' data through the gateway, controller, and platform channels connected at the edge, fog, and cloud levels, respectively. A doctor may have its own request format, and each sensor may have a different response format.

In the first case (depicted with blue edges and vertices), the doctor D1 is attached to the gateway G1 and wants to access data of bed 1 patient admitted to the ICU of hospital 1. The request is sent to the sensors and actuators attached to the patient bed in MQTT format. The sensors, in this case, are HBRM and GLM, and the actuators are SID and AID. The data size detail of these devices is given in Table 4. In this scenario, the data travel a single hop. The communication channel demand for this case is calculated in Table 5 using (16) and is determined as 56 bytes.

Now, take the other scenario where a doctor D2 is attached to the controller at fog level and wants to access the data of patient bed 2 and patient bed 3 of hospital 1. The healthcare device sent the data to the controller in CoAP format and forwarded it to relevant gateways. Detail of attached sensors/actuators and response size is given in Table 4. As given in Table 6, the maximum communication channel demand for hospital 1 is calculated as 64 bytes and for hospital 2 and hospital 3, it is calculated to be 95 bytes each.

The next scenario of ICCD assessment is where the doctor D3 is linked to the cloud platform and wants to access the data of different patients admitted to different hospitals. The request is forwarded to the sensor through the platform to the controller to the gateway down to the sensors. The request format is AMPQ with 8 bytes. By using the details given in Table 7, Figure 8, and (16), the maximum communication channel demand for this scenario is determined to be 95B. It is the largest amount of data that can transfer over an interaction which is determined by the maximum interaction weight value among all of these interaction weights.

5.5. IoT Interoperability Resolving Network Complexity (IIRNC) and Interoperability Resolving Network Load (IIRNL) Evaluation. Consider a scenario where a doctor initiates a request to the gateway G1 devices for accessing patient data in JSON format. The gateway retrieves the data from devices and checks its format. If the format is as required, the data are transferred to the health care doctor's device. However, if the data format translation is required, the gateway sends the data to the fog layer, which transfers the data to the cloud for

Table 3: Data format and packet sizes for low-end medical devices considered in the case study.

Sensor/Actuators	Data generation Time interval	Protocol Supported	Data Format	Packet size (In bytes)
GLM	5 minutes	CoAP = 4BYTES	TEXT	8
HBRM	1 minute	REST = 8BYTES	JSON	12
BPM	1 minute	MQTT = 2BYTES	Number	7
RRM	1 minute	CoAP = 4BYTES	XML	7
POM	5 minutes	REST = 8BYTES	TEXT	12
ECG	1 minute	MQTT = 2BYTES	JSON	10
TMS	1 minute	MQTT = 2BYTES	NUMBER	6
CMS	1 minute	MQTT = 2BYTES	NUMBER, GRAPH	8
IS	1 minute	MQTT = 2BYTES	JSON	20
CSI	1 minute	CoAP = 4BYTES	XML	16
FDS	1 minute	AMQP = 8 BYTES	JSON	23
SID	5 minute	CoAP = 8BYTES	XML	20
SV	5 minute	CoAP = 8BYTES	XML	24
AID	60 minutes	AMQP = 8BYTES	JSON	16
SR	CONDITIONAL	AMQP = 8BYTES	JSON	

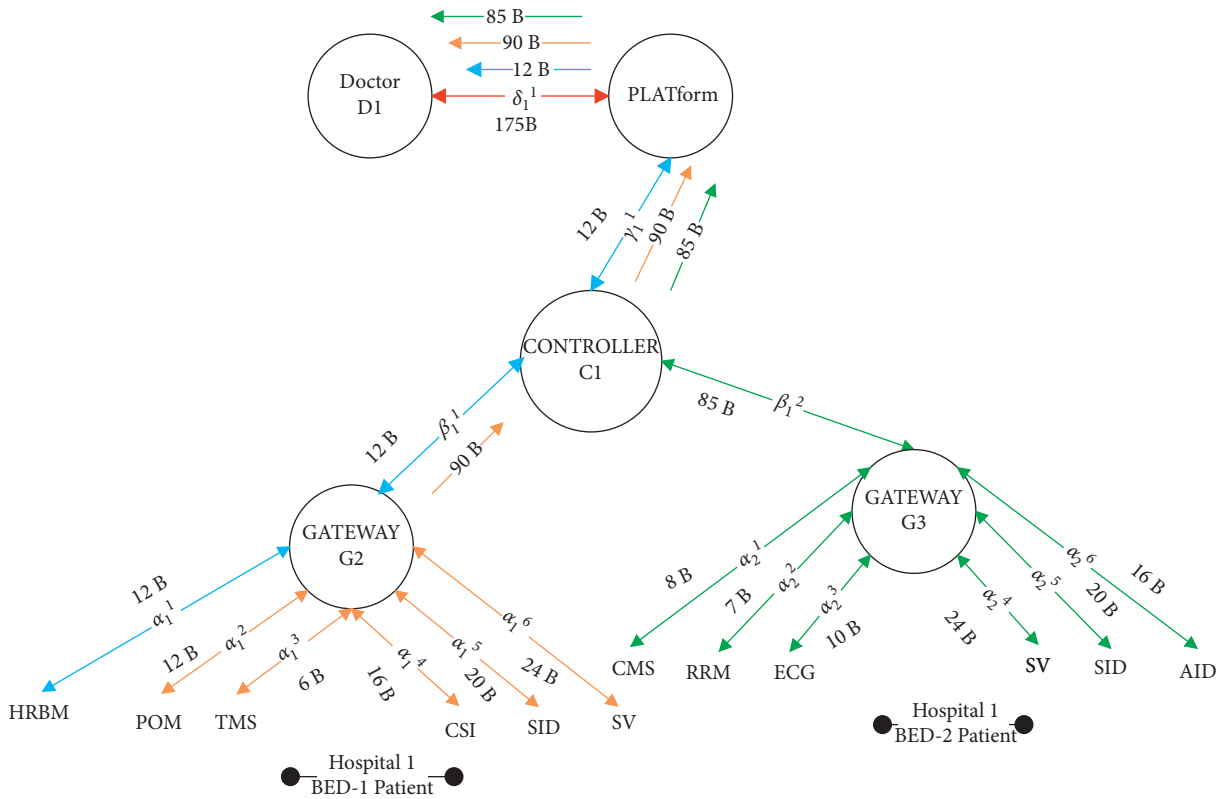

Figure 7: Data loads for scenario (i), scenario (ii), and scenario (iii) of the smart healthcare IoT system for IoT transactional network load calculations.

translation. The translated data are sent back to the gateway through the fog layer and then sent to the doctor. The translation scenario is presented in Figure 9. The IoT interoperability resolving network complexity and load is determined by hop count and the weight of the edges involved. In the scenario given in Figure 9, four hops are traversed when the interoperability is not required, so the interoperability network complexity is 4, and the interoperability network load is 78 bytes. However, the data translation is required for interoperability when the doctor needs data in JSON format and his device supports CoAP format. This scenarios is given in Figure 8 for patient at bed-2 of hospital-1. The bed 2 has devices BPM, POM, TMS, SID, AID, and SV and is supporting data format Number/binary, text, number/binary, XML, JSON, and XML, respectively. The protocols supported are MQTT, REST, and AMQP as

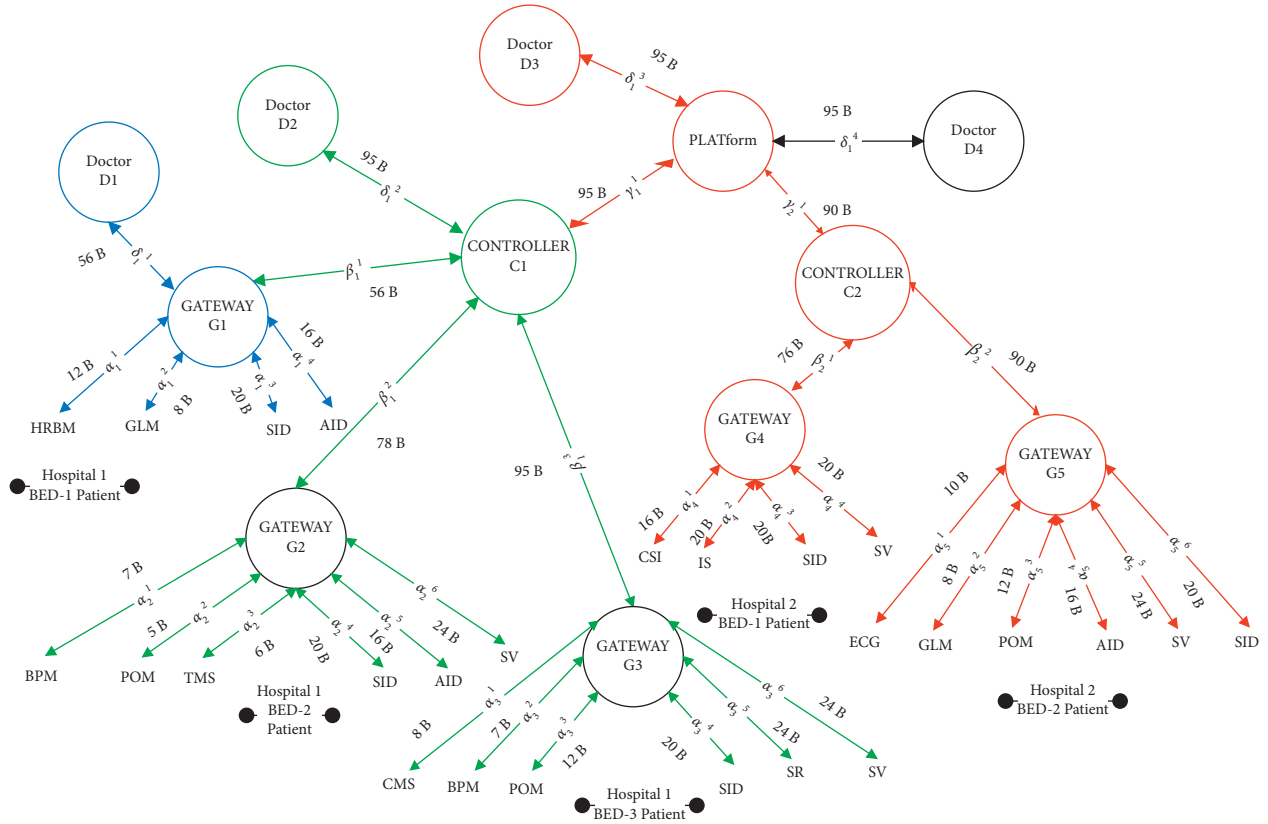

Figure 8: Evaluation of IoT communication channel demand scenario (i): blue, scenario (ii): blue and green, and scenario (iii): red, blue, and green.

Table 4: Response size of actuators/sensors attached to patients bed.

Evaluation cases	Hospital 1			Hospital 2	
	Bed 1	Bed 2	Bed 3	Bed 1	Bed 2
Sensors	HRBM, GLM	BPM, POM, TMS	CMS, BPM, POM	CSI, ID	ECG, GLM, POM
Actuators	SID, AID	SID, AID, SV	SID, SR, SV	SID, SV	AID, SV, SID
Response-size					
Sensor	12B, 8B	7B, 5B, 6B	8B, 7B, 12B	16B, 20B	10B, 8B, 12B
Actuator	20B, 16B	20B, 16B, 24B	20B, 24B, 24B	20B, 20B	16B, 24B, 20B

Table 5: Data channel requirement where doctor D1 access data through gateway.

Sensors	Nodes	Edge name	ReqSize	ResSize (B)
HRM	Low-end device-gateway 1	Alpha	MQTT = 2BYTES	12
GLM	Low-end device-gateway 1	Alpha	MQTT = 2BYTES	8
SID	Low-end device-gateway 1	Alpha	MQTT = 2BYTES	20
AID	Low-end device-gateway 1	Alpha	MQTT = 2BYTES	16
Accumulated	gateway1-doctor 1	Delta		56

given in Table 3. After interoperability, the involved network's complexity using (19) is calculated as 8, and the network load introduced is 69 bytes.

5.6. Discussion. The basic aim of this research was to present measurement criteria for quantitatively assessing the design quality of IoT systems and services. To achieve our goal, we defined a suite of design quality metrics for IoT systems evaluation. We take the IoT EcoSystem as a base of this work and drafted the IoT EcoSystem model as an abstract graph to define the IoT transactions model in terms of schema graph. The formal definitions are used to define the design quality metrics for IoT services quality assessment from its design models. The metrics defined are IoT design complexity metric, IoT transactional complexity metric, IoT

Table 6: ICCD evaluation for doctor D2, who accesses the data of multiple patients admitted in hospital 1. The request format in this case is CoAP with 2 bytes.

Evaluation scenario	Nodes	Edge name	ICCD	ICDD (B)
Hospital 1 Bed 1	Low-end devices-gateway G1	Alpha	Max (α)	20
	Controller C1 gateway G1-	Beta	Max $(\sum \alpha_x^y)$	56
	Controller C1-doctor D2	Gamma	Max $(\beta_1^1, \beta_2^1, \beta_3^1)$	64
Hospital 1 Bed 2	Low-end devices-gateway G2	Alpha	Max (α)	24
	Controller C1 gateway G2-	Beta	Max $(\sum \alpha)$	78
	Controller C1-doctor D2	Gamma	Max $(\beta_1^1, \beta_2^1, \beta_3^1)$	95
Hospital 1 Bed 3	Low-end devices-gateway G3	Alpha	Max (α)	24
	Controller C1 gateway G3-	Beta	Max $(\sum \alpha)$	95
	Controller C1-doctor D2	Gamma	$\max(\beta_1^1, \beta_2^1, \beta_3^1)$	95

Table 7: Evaluation of ICCD where a remote doctor (D3 or D4) accesses data of patients admitted in different hospitals through cloud and uses the request format of AMPQ = 8BYTES.

Evaluation scenario	Nodes	Edge name	ICCD edge	ICDD value
Hospital 1 Bed 1	Low-end devices-gateway1	Alpha	Max (α)	20B
	Gateway G1-controller C1	Beta	Max $(\sum \alpha)$	56B
	Controller C1-platform P1	Gamma	Max $(\beta_1^1, \beta_2^1, \beta_3^1)$	64B
	Platform P1-doctor-D3	Delta	Max (γ_1^1, γ_2^1)	56B
Hospital 1 Bed 2	Low-end devices-gateway2	Alpha	Max (α)	24B
	Gateway G1-controller C1	Beta	Max $(\sum \alpha)$	78B
	Controller C1-platform P1	Gamma	Max $(\beta_1^1, \beta_2^1, \beta_3^1)$	95B
	Platform P1-doctor-D3	Delta	Max (γ_1^1, γ_2^1)	95B
Hospital 1 Bed 3	Low-end devices-gateway3	Alpha	Max (α)	24B
	Gateway G3-controller C1	Beta	Max $(\sum \alpha)$	95B
	Controller C1-platform P1	GAMMA	Max $(\beta_1^1, \beta_2^1, \beta_3^1)$	95B
	Platform P1-doctor-D3	Delta	Max (γ_1^1, γ_2^1)	95B
Hospital 2 Bed 1	Low-end devices-gateway4	Alpha	Max (α)	20B
	Gateway G4-controller C2	Beta	Max $(\sum \alpha)$	76B
	Controller C2-platform P1	GAMMA	Max (β_1^2, β_2^2)	90
	Platform P1-doctor-D3	Delta	Max (γ_1^1, γ_2^1)	95B
Hospital 2 Bed 2	Low-end devices-gateway4	Alpha	Max (α)	24B
	Gateway G5-controller C2	Beta	Max $(\sum \alpha)$	90B
	Controller C2-platform P1	GAMMA	Max (β_1^2, β_2^2)	90B
	Platform P1-doctor-D3	Delta	Max (γ_1^1, γ_2^1)	95B

transactional network load, IoT communication channel demand, IoT interoperability resolving network complexity, IoT interoperability resolving network load, and IoT reusability factor. The assessment of defined design quality metrics is made with a case study of the smart healthcare IoT system. We evaluated the proposed metrics using different scenarios of the smart healthcare system. The calculated results are given in tabular form which presents the metrics outcomes to be compared with realistic outcomes. The metrics results are proved validated against benchmark values taken from field experts. It validates the accuracy of defined metrics as the obtained values are consistent with known parameters (i.e., predicted results). The results are useful in comparing the design quality of different models designed for an IoT service. Using these metrics, we can compare different designs available for an IoT system or its services. The best service model will be the one having minimum value for the metrics applied for evaluation.

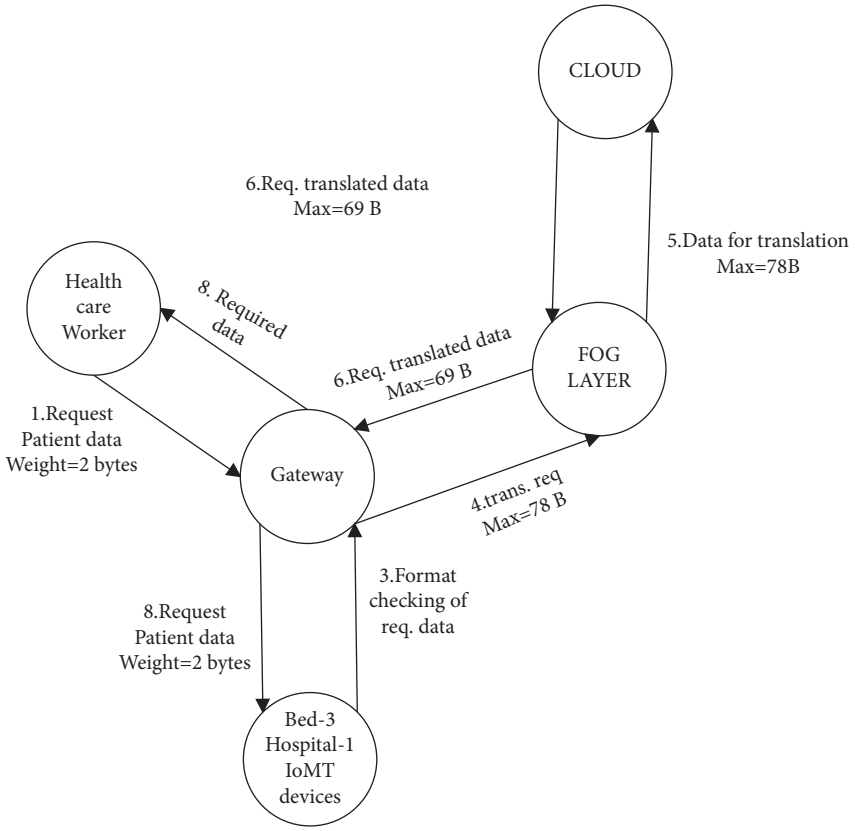

Figure 9: IIRNL-IoT interoperability resolving network load metric calculation with a use case of healthcare IoT-enabled smart city.

6. Conclusion

This work presented metrics for design quality evaluation of IoT services by defining the IoT EcoSystem as a schema Graph. The metrics defined for different IoT parameters measure design complexity, transactional complexity, transactional network load, communication channel demand, interoperability resolving network complexity, interoperability resolving network load, and reusability. The quality metrics have been mapped to the quality factors of the IoT system, including efficiency, portability, compatibility, maintainability, usability, and functional suitability according to IoT characteristics. The IoT quality factors have been mapped to IoT characteristics which include heterogeneity, resource-constrained devices, collaboration with hardware and resources, network mobility, embedded and adaptive devices, and design to monitor IoT devices. The QoS factors are then mapped to IoT services which are divided into information aggregation, identity-related, ubiquitous, and collaborative-aware services. The defined metrics are evaluated with different use cases of a smart IoT healthcare system. Finally, we conclude that the proposed design quality metrics can be used to compare and evaluate design level quality for the quality of services in IoT systems. This research work can be automated in future work, and the metrics can be directly computed to save validation time. These proposed metrics can also be used in different conditions to measure the accuracy and correctness of an IoT project.

Data Availability

Data are available on request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Authors' Contributions

The authors contributed equally to the design of ideas, analysis of results, and writing of the article.

References

[1] M. S. A. S. Shazia and F. Majeed, "Software design quality metrics for web based applications," *Pakistan Journal of Science*, vol. 63, no. 1, 2011.

[2] M. Gigli and S. Koo, "Internet of things: services and applications categorization," *Advances in Internet of Things*, vol. 1, no. 2, pp. 27–31, 2011.

[3] M. Bures, X. Bellekens, K. Frajtak, and B. S. Ahmed, "A comprehensive view on quality characteristics of the iot solutions," in *EAI International Conference on IoT in Urban Space*Springer, Manhattan, NY, USA, 2018.

[4] G. White, V. Nallur, and S. Clarke, "Quality of service approaches in iot: a systematic mapping," *Journal of Systems and Software*, vol. 132, pp. 186–203, 2017.

[5] B. Costa, P. F. Pires, F. C. Delicato, W. Li, and A. Y. Zomaya, "Design and analysis of iot applications: a model-driven approach," in *Proceedings of the 2016 IEEE 14th Intl Conf on*

Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), pp. 392–399, IEEE, Auckland, New Zealand, August 2016.

[6] A. Behura, A. Narayan, A. Ray, and S. K. Pani, "A complete model for iot application," in *Proceedings of the 2017 International Conference on Intelligent Sustainable Systems (ICISS)*, pp. 1026–1030, IEEE, Palladam, India, December 2017.

[7] S. Bansal and D. Kumar, "Iot ecosystem: a survey on devices, gateways, operating systems, middleware and communication," *International Journal of Wireless Information Networks*, vol. 27, no. 3, pp. 340–364, 2020.

[8] A. S. Gillis, "What is iot (internet of things) and how does it work?," 2021, https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT.

[9] F. C. Delicato, P. F. Pires, T. Batista, E. Cavalcante, B. Costa, and T. Barros, "Towards an iot ecosystem," in *Proceedings of the First International Workshop on Software Engineering for Systems-Of-Systems*, pp. 25–28, Montpellier, France, July 2013.

[10] L. Li, S. Li, and S. Zhao, "QoS-aware scheduling of services-oriented internet of things," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1497–1505, 2014.

[11] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: a survey on enabling technologies, protocols, and applications," *IEEE communications surveys & tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[12] J. Kiruthika and S. Khaddaj, "Software quality issues and challenges of internet of things," in *Proceedings of the 2015 14th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)*, pp. 176–179, IEEE, Guiyang, China, August 2015.

[13] M. Kim, "A quality model for evaluating iot applications," *International Journal of Computer and Electrical Engineering*, vol. 8, no. 1, pp. 66–76, 2016.

[14] J. J. Tambotoh, S. M. Isa, F. L. Gaol, B. Soewito, and H. L. H. S. Warnars, "Software quality model for internet of things governance," in *Proceedings of the 2016 International Conference on Data and Software Engineering (ICoDSE)*, pp. 1–6, IEEE, Denpasar, Indonesia, October 2016.

[15] G. Tanganelli, C. Vallati, and E. Mingozzi, "Ensuring quality of service in the internet of things," in *New Advances in the Internet of Things*, vol. 715, Manhattan, NY, USA, Springer, 2018.

[16] M. Singh and G. Baranwal, "Quality of service (qos) in internet of things," in *Proceedings of the 2018 3rd International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, pp. 1–6, IEEE, Bhimtal, India, Feburary 2018.

[17] M. Bures, T. Cerny, and B. S. Ahmed, "Internet of things: current challenges in the quality assurance and testing methods," in *International Conference on Information Science and Applications*Springer, Manhattan, NY, USA, 2018.

[18] L. E. V. Zavala, A. O. García, and M. Siller, "Architecture and algorithm for iot autonomic network management," in *Proceedings of the 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pp. 861–867, IEEE, Atlanta, GA, USA, July 2019.

[19] M. Suryanegara, D. A. Prasetyo, F. Andriyanto, and N. Hayati, "A 5-step framework for measuring the quality of experience (qoe) of internet of things (iot) services," *IEEE Access*, vol. 7, pp. 175 779–175 792, 2019.

[20] A. Jaleel, T. Mahmood, M. A. Hassan, G. Bano, and S. K. Khurshid, "Towards medical data interoperability through collaboration of healthcare devices," *IEEE Access*, vol. 8, Article ID 132302, 2020.

[21] F. E. F. Samann, S. R. M. Zeebaree, and S. Askar, "Iot provisioning qos based on cloud and fog computing," *Journal of Applied Science and Technology Trends*, vol. 2, no. 1, pp. 29–40, 2021.

[22] M. O. Thomas and B. B. Rad, "Reliability evaluation metrics for internet of things, car tracking system: a review," *International Journal of Information Technology and Computer Science*, vol. 9, no. 2, pp. 1–10, 2017.

[23] H.-W. Jung, S.-G. Kim, and C.-S. Chung, "Measuring software product quality: a survey of iso/iec 9126," *IEEE software*, vol. 21, no. 5, pp. 88–92, 2004.

[24] S. De, F. Carrez, E. Reetz, R. Tönjes, and W. Wang, "Test-enabled architecture for iot service creation and provisioning," in *The Future Internet Assembly*Springer, Manhattan, NY, USA, 2013.

[25] I. Yaqoob, E. Ahmed, I. A. T. Hashem et al., "Internet of things architecture: recent advances, taxonomy, requirements, and open challenges," *IEEE Wireless Communications*, vol. 24, no. 3, pp. 10–16, 2017.

[26] R. Zhohov, "Evaluating Quality of Experience and Real-Time Performance of Industrial Internet of Things," Lappeenranta University of Technology, Lappeenranta, Finland, Degree Program in Computer Science, 2018.

[27] G. Gardašević, M. Veletić, N. Maletić et al., "The iot architectural framework, design issues and application domains," *Wireless Personal Communications*, vol. 92, no. 1, pp. 127–148, 2017.

[28] M.-A. Nef, L. Perlepes, S. Karagiorgou, G. I. Stamoulis, and P. K. Kikiras, "Enabling qos in the internet of things," in *Proceedings of the 5th Int. Conf. on Commun., Theory, Reliability, and Quality of Service (CTRQ 2012)*, pp. 33–38, Citeseer, France, April 2012.

[29] A. Al-Qerem, M. Alauthman, A. Almomani, and B. B. Gupta, "Iot transaction processing through cooperative concurrency control on fog–cloud computing environment," *Soft Computing*, vol. 24, no. 8, pp. 5695–5711, 2020.

[30] T. J. McCabe and C. W. Butler, "Design complexity measurement and testing," *Communications of the ACM*, vol. 32, no. 12, pp. 1415–1425, 1989.

[31] S. R. Chidamber and C. F. Kemerer, "A metrics suite for object oriented design," *IEEE Transactions on Software Engineering*, vol. 20, no. 6, pp. 476–493, 1994.

[32] ICT Express, *Autonomic interoperability manager: A Service-Oriented Architecture for Full-Stack Interoperability in the Internet-Of-Things*, ICT Express, Birmingham, UK, 2021.