International Journal of Smart Device and Appliance
Vol.8, No.1 (2020), pp.17-24
http://dx.doi.org/10.21742/ijsda. 2020.8.1.03

Design of Context-Aware Resource Management for Healthcare IoT

Siwoo Byun

Anyang University, Kyoungkido, South Korea
swbyun30@daum.net

Abstract

Edge computing is emerging as a technology that complements cloud computing in an IoT environment where huge amounts of data are generated in real time. This paper introduces recent IoT network and edge computing technology and describes healthcare IoT network and related edge technologies. This paper also proposes efficient resource management called CaRM to ensure stable data services for sensor nodes and edge gateways in the edge-based IoT environment. In CaRM, the context-aware classification and dynamic resource management are devised to improve the stability and performance for healthcare IoT service. Based on the context-aware classification, the sensor-level resource tree and application-level resource tree are generated. After creating a service-context tree that incorporates these two trees, CaRM controls the permitted range of individual resource consumption at each service level to guarantee ensure service.

Keywords: *Internet of things, Healthcare service, Context-aware classification*

1. Introduction

Advances in computing and networking technologies have made extensive use of small information sensor devices. Each information device could include its management tools and a small database. Flash memory is one of the best candidates for data management in ubiquitous computing environments.

Sensor database is in integral component of the increasing reality of the Internet of Things (IoT) environment [1][2]. Much of the data transmitted is sensor data. The huge volume of data produced and transmitted from sensing devices can provide a lot of information but is often considered the next big data challenge for businesses. Recently, various sensor database systems, including TinyDB and Cougar [3], have been developed for the efficient management of sensor data in the USN environment.

Most IoT applications are data-centric since sensor nodes are designed from the point of measured data rather than identified data such as ip address of conventional networks. That is, measured data is most important in sensor networks. From this architectural point of view, sensor network is treated as a huge database called sensor database [4].

Recent column-based storage model [5][6][7] is more advantageous than general storage models for storing sensor data. A column-based data storage store data in the order of columns and not in the order of rows(records) as in general data storages. Especially, it is more I/O efficient for read-only queries, such as sensor queries, because it only accesses the columns (or attributes) required for the queries.

Article history:
Received (March 28, 2020), Review Result (May 2, 2020), Accepted (June 5, 2020)

Print ISSN: 2288-8977, eISSN: 2207-404X IJSDA

2. Related works

IoT edge computing [8][9][10] is defined as a distributed computing infrastructure that includes a large number of sensor devices that are well connected to each other. It is a new paradigm that provides high computing resources close to distributed IoT devices [Figure 1]. Therefore, IoT edge node can collect, classify, and analyze raw data streams locally instead of transferring them to the remote cloud, significantly reducing traffic overheads and speeding up the IoT big data process. However, where IoT edge nodes should be placed to facilitate communication between IoT devices and the edge nodes is still an important issue.

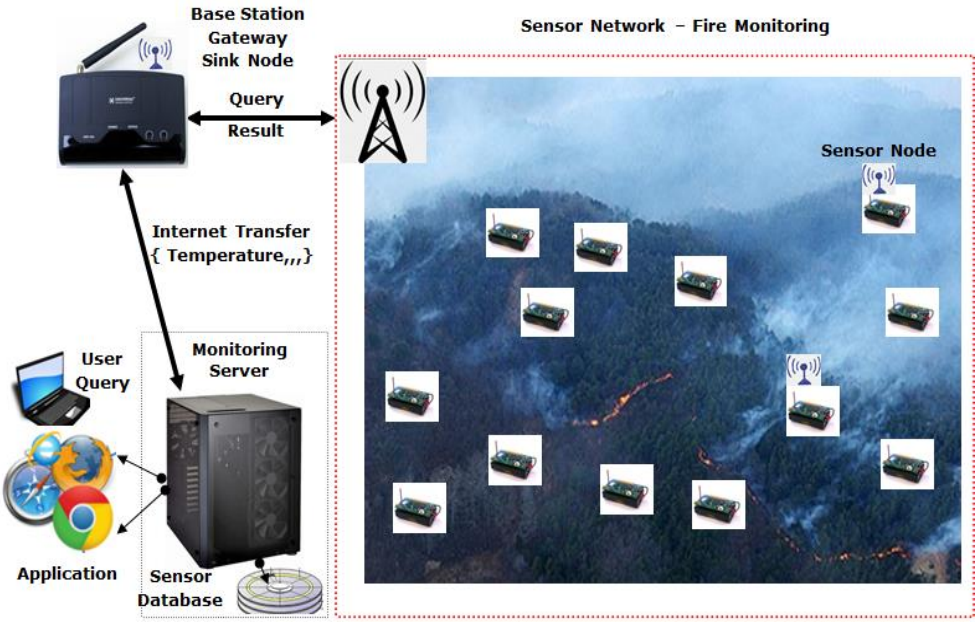

Figure 1. Example of IoT sensor environment

[Figure 1] is an example of fire monitoring applications. Sensor nodes collects fire-related signals such as temperature, wind speed and humidity, and sends them to IoT gateways such as base station. The gateway handles periodic sensor data collection from the sensor nodes, local information processing such as offloading and filtering. The gateway also serve as relay agent to the fire monitoring server in the cloud environment. The server collects meaningful data among them from the gateways, analyzes them for fire monitoring.

IoT-based edge computing is rapidly rising, and medical edge environment is an important IoT field. There are many different things to do in the medical IoT environment. For example, medical information monitoring, control of medical actuators, management of medical sensor devices, device communication, remote care, and medical big data analysis are required. The following example is a brief representation of the medical IoT edge environment [Figure 2].

In this medical environment, edge gateways play a central role in edge computing, and information should be output without delay to the medical terminals in [Figure 3]. Therefore, in most cases, predictable and fast responses are required. Examples that require a quick response include adjusting the frequency of electrical impulses based on heart rate or adjusting the insulin release rate in automatic pumps based on blood glucose and other vital signs.

Streaming a patient's medical signals to a control panel in real time is also a delay sensitive case. Using local processing power and local networking, medical gateways can stream real-

time signals such as critical ECG (electrocardiograms) and artificial imaging to IoT tablets without relying on Internet connectivity.

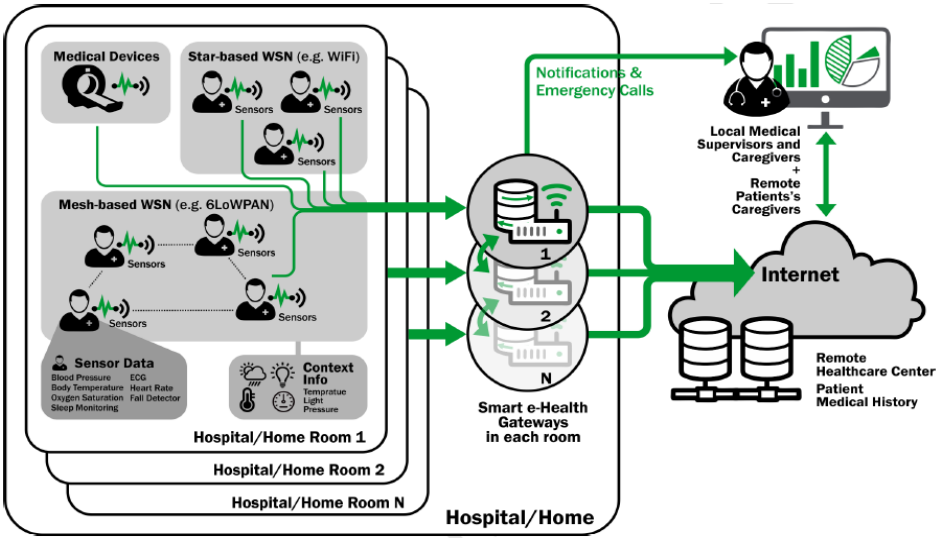

Figure 2. IoT Edge Computing in medical environments [11]

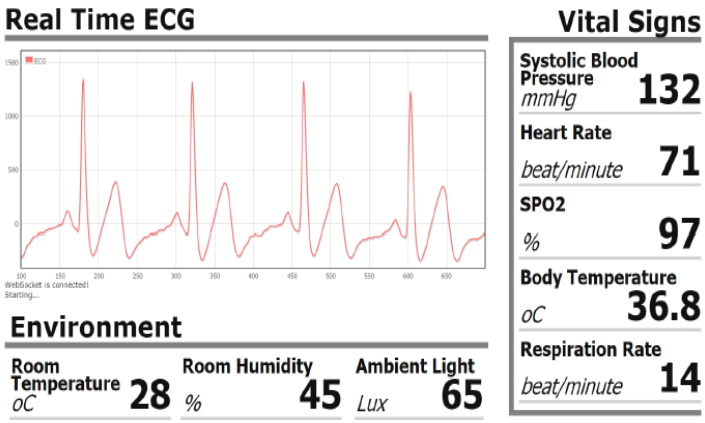

Figure 3. Display of medical tablet

If the vital signs to be provided for this client terminal are not displayed, serious problems arise for both the patient and the medical institution. Compared to cloud servers, which can send notifications in several ways, gateways can only transmit vital data through limited media. However, even if the cloud server is unavailable, it is possible to independently transmit critical data around the gateway within the local network. In other words, the edge gateway architecture maximizes the reliability of the health care system and enables users to receive critical vital signs on time [11].

However, the gateway must also store the received data in the local store. For secure data services, edge gateways use non-volatile memory, such as flash memory, as a local repository.

Edge gateways can store data on local storage differently depending on the type and importance of health care. In addition, medical data can be compressed or encrypted to export to higher systems, such as medical cloud. Thus, edge gateways require efficient data management to perform critical functions such as data analysis, compression, filtering, encryption, and so on [6].

The transmission from edge gateway to the cloud is limited by network bandwidth, and data computations are limited by CPU power. In case of uneven data transfer, local storage should also act as a cache to maintain a continuous data stream. That is, edge gateways require techniques to maintain a stable health care service even if the network is unstable [12].

3. Resource management for IoT edge computing

In this study, we propose efficient resource management techniques to ensure stable data service for sensor node and edge gateway in IoT edge computing environments. That is, to achieve the goal of guaranteed service reliability for transmission, storage, and processing data, a context-aware resource management (CaRM) scheme is proposed.

To support efficient and stable service of the IoT data, the characteristics of the sensor node and the level of service assurance must be analyzed in advance. Furthermore, data generated continuously from a lot of sensor nodes is very large volume. If all the sensor data are delivered to the cloud server, this work load prevents critical sensor data from being processed in real time. Thus, local processing of edge gateway can increase service responsiveness and data reliability. Smart offloading techniques can eliminate unnecessary information in advance, so that only essential information is delivered to the cloud. Moreover, the battery consumption of individual sensor nodes can be also reduced.

3.1. Classification of IoT service levels

Although the proposed CaRM scheme is applicable to general IoT, we focus on its use in medical IoT for easy understanding. We also consider all the resources associated with the quality of medical services as one integrated resource which includes communication bandwidth, storage memory, CPU resources, precision of sensor values, and frequency of sensor value transmission. In order to enhance service efficiency and reliability, these resources need to be well classified before actual use.

First, CaRM classifies all data services into different classes based on delay-sensitive levels. This classification applies to all sensor nodes connected to the medical gateways that serve as the core of medical IoT Edge. Secondary classification includes service priorities, traffic type, user type, and the rate of change of sensor values.

In the general environment, resource managers distribute computing resources in order, usually in first-in first-out (FIFO) mode, except in special cases. However, in a medical environment, there is a great need to prepare for emergencies and operate in a mode that specializes in urgent medical tasks (intensive care, patient monitoring) that deals with life. In particular, in emergency mode, the exclusive priority of task handling should be enhanced, while common sensor signals that are non-critical should be delayed or dropped.

Based on these criteria, services are classified in detail, and related properties and methods are initially set. However, if there is less variation over a long period of time in the medium-term setting, the transmission cycle can be increased to the long-term setting. The patients in dangerous groups are given higher priorities. However, they have higher priority and short-term transmission cycle when sudden changes in critical sensor values occur, even though they are ordinary patients.

3.2. Context-based resource tree

For efficient management of service resources, CaRM establishes the basic properties, analyzes the relevant characteristics, and creates several characteristic-trees. The individual characteristic-trees are then integrated to one resource-tree. Finally, based on this integrated tree, real-time dynamic resource management is carried out.

The following example is 'Sensor-Level Resource Weight Tree,' the first characteristic-tree suitable for medical IoT environments. The characteristics of each sensor are based on the medical tablet device in [Figure 3]. The combination of characteristic-trees ultimately results in an integrated tree reflecting the resource weights.

In this study, an integrated resource such as wired and wireless bandwidth, memory, and CPU are called 'IoT resource'. [Figure 4] is an example of a sensor-level resource weight tree that shows the percentage of IoT resources to be distributed for each sensor nodes. CaRM allocates 25% ECG service resource, 20% heart rate, 15% blood pressure, and 15% body temperature.

The selections of IoT resources was referenced in [12]. The sensor selection for ECG service was referenced in [11] and the above weight distribution for heart rate, blood pressures, body temperature is predefined for simplicity, but can be changed as needed.

The level sensor of the battery in the medical terminal is 10%, and the status of the patient's room is 15% (room temperature sensor, humidity sensor, light sensor). CaRM performs resource management tasks by referring to the settings of this resource tree.

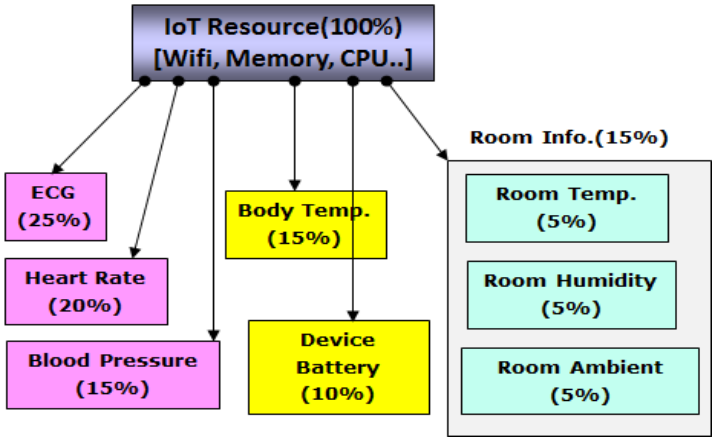

Figure 4. Sensor-level resource weight tree

The following example is the resource partitioning by application using the service [Figure 5]. Each class uses resources as a proportion assigned to it. When resources are scarce, a minimum is guaranteed according to the predefined priorities.

Based on the resource tree in [Figure 4] and [Figure 5], an integrated tree called Service-context Tree can be created that reflects the sensor-level weight and application-level weight. [Figure 6] shows service-context tree with detailed weights to be serviced for intensive care. In this way the overall integrated resource tree for medical IoT can be created.

In order to guarantee reliable service based on the integrated tree, efficient methods to ensure that critical resources are kept to a minimum should also be considered in the event of a shortage of resources. Since the most heavily congested resource is the wireless network

channel, CaRM manages wireless related information in the network QoS module. QoS module maintains Minimum Guaranteed Bandwidth (MGB) with service priority so that urgent medical data can be communicated during heavy network load.

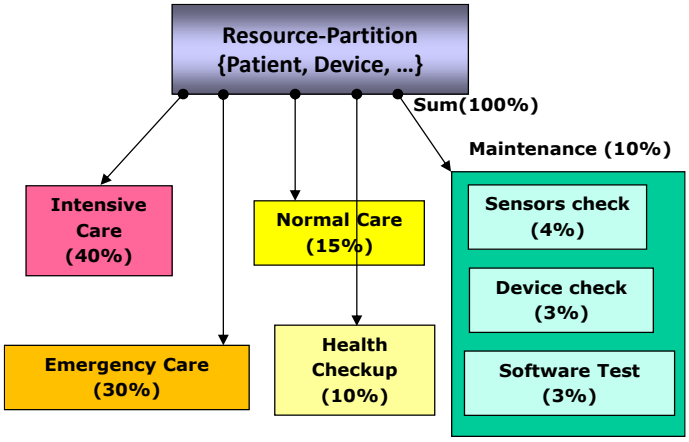

Figure 5. Application-level resource weight tree

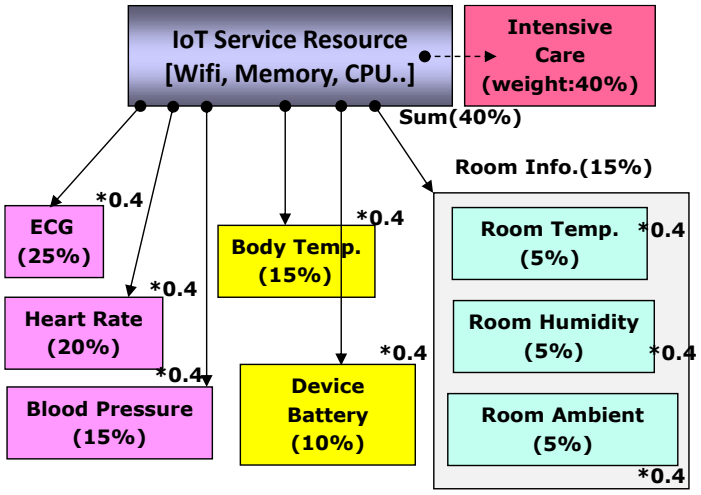

Figure 6. Service-context tree

In addition, CaRM set the type of data to be transferred from edge gateways to top cloud servers and set up the transmission cycle. This means that all sensor data does not have to be sent to the cloud, nor do all sensor data need to be sent at the same cycle.

4. Conclusions

This study introduced IoT network technology and the recent edge computing technology. As a representative case, medical IoT network was described and related edge technologies were analyzed. This paper also proposed efficient resource management called CaRM to ensure stable data services for sensor nodes and edge gateways in the edge-based IoT environment.

In CaRM, the context-aware resource management were proposed to improve the stability and performance for medical IoT service. Based on the context-aware classification, the sensor-level resource tree and application-level resource tree were generated. After creating a

service-context tree that incorporates these two trees, CaRM controls the permitted range of individual resource consumption.

References

[1] R. Govindan, J. M. Hellerstein, W. Hong, S. Madden, M. Franklin, and S. Shenker, "The sensor network as a database," https://www.ics.uci.edu/~dsm/ics280sensor/readings/data/02-771.pdf, April 25 **(2020)**

[2] P. Bonnet, J. Gehrke, and P. Seshadri, "Towards sensor database systems," The 2nd International Conference on Mobile Data Management," Hong Kong, January 8-10, **(2001)**

[3] https://tinydb.readthedocs.io/en/latest/, June.25 **(2020)**

[4] S. Yeon and J. Park, "IoT platform analysis and issues," ETRI Insight Report, vol.27, pp.1-56, **(2016)**

[5] D.S. Kim, J.S. Kim, B.J. You, and H.K. Jung, "Efficient dynamic index structure for SSD (SPM)," Journal of Korea Contents Association, vol.10, no.2, pp.54-62, **(2010)**

[6] S. Byun and S. Jang, "Asymmetric index management scheme for high-capacity compressed databases," Journal of Korea Academia-Industrial, vol.17, no.7, pp.293-300, **(2016)**

[7] S. Ahn and K. Kim, "A join technique to improve the performance of star schema queries in column-oriented databases," Journal of Korean Institute of Information Scientist and Engineers, vol.40, no.3, pp.209-218, **(2013)**

[8] S. Yi, Z. Qin, and Q. Li, "Security and privacy issues of fog computing: A survey," in International Conference on Wireless Algorithms, Systems and Applications (WASA), **(2015)**

[9] Shanhe Yi, Zijiang Hao, Zhengrui Qin, and Qun Li, "Fog computing: platform and applications," The 3rd IEEE Workshop on Hot Topics in Web Systems and Technologies, Washington, DC, USA, November 12-13, **(2015)**

[10] Gopika Premsankar, Mario Di Francesco, and Tarik Taleb, "Edge computing for the internet of things: a case study," ieee internet of things journal, vol.5, no.2, pp.1275-1284, **(2018)**

[11] Amir M. Rahmani and Tuan Nguyen Gia, "Behailu Negash,Arman Anzanpour, Iman Azimi, Mingzhe Jiang, Pasi Liljeberg, Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things: A fog computing approach, Future Generation Computer Systems," vol.78, no.2, pp.641-658, **(2018)**

[12] S. Byun, "Gateway-based resource control for reliable iot environments" International Journal of Advanced Trends in Computer Science and Engineering, vol.8, no.5, pp.1881-1885, **(2019)**

This page is empty by intention.