Future Generation Computer Systems 78 (2018) 641–658

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things: A fog computing approach

Amir M. Rahmani [a,b,*], Tuan Nguyen Gia [c], Behailu Negash [c], Arman Anzanpour [c], Iman Azimi [c], Mingzhe Jiang [c], Pasi Liljeberg [c]

[a] Department of Computer Science, University of California Irvine, USA
[b] Institute of Computer Technology, TU Wien, Vienna, Austria
[c] Department of Information Technology, University of Turku, Turku, Finland

ARTICLE INFO

Article history:
Received 26 May 2016
Received in revised form 8 November 2016
Accepted 8 February 2017
Available online 10 February 2017

Keywords:
Internet of Things
Healthcare
Edge/Fog computing
Mobility
Smart hospital
Home care
Smart gateway
Sensor network

ABSTRACT

Current developments in ICTs such as in Internet-of-Things (IoT) and Cyber–Physical Systems (CPS) allow us to develop healthcare solutions with more intelligent and prediction capabilities both for daily life (home/office) and in-hospitals. In most of IoT-based healthcare systems, especially at smart homes or hospitals, a bridging point (i.e., gateway) is needed between sensor infrastructure network and the Internet. The gateway at the edge of the network often just performs basic functions such as translating between the protocols used in the Internet and sensor networks. These gateways have beneficial knowledge and constructive control over both the sensor network and the data to be transmitted through the Internet. In this paper, we exploit the strategic position of such gateways at the edge of the network to offer several higher-level services such as local storage, real-time local data processing, embedded data mining, etc., presenting thus a Smart e-Health Gateway. We then propose to exploit the concept of Fog Computing in Healthcare IoT systems by forming a Geo-distributed intermediary layer of intelligence between sensor nodes and Cloud. By taking responsibility for handling some burdens of the sensor network and a remote healthcare center, our Fog-assisted system architecture can cope with many challenges in ubiquitous healthcare systems such as mobility, energy efficiency, scalability, and reliability issues. A successful implementation of Smart e-Health Gateways can enable massive deployment of ubiquitous health monitoring systems especially in clinical environments. We also present a prototype of a Smart e-Health Gateway called UT-GATE where some of the discussed higher-level features have been implemented. We also implement an IoT-based Early Warning Score (EWS) health monitoring to practically show the efficiency and relevance of our system on addressing a medical case study. Our proof-of-concept design demonstrates an IoT-based health monitoring system with enhanced overall system intelligence, energy efficiency, mobility, performance, interoperability, security, and reliability.

1. Introduction

Internet of Things (IoT) is getting a wide acceptance and a growing adoption in many aspects of our daily life [1,2]. IoT technology provides a competent and structured approach to improve health and wellbeing of mankind. It is predicted that IoT-based systems will remodel the healthcare sector in terms of social benefits and penetration as well as cost-efficiency [3,4]. Due to the ubiquitous computing nature of IoT, all the healthcare system entities (individuals, appliances, medicine) can be monitored and managed continuously. By applying IoT technologies to healthcare, the quality and cost of medical care can be improved by automating tasks previously performed by humans [5–7]. In that sense, IoT enables Electronic Health (eHealth), Mobile Health (mHealth) and Ambient Assisted Living (AAL) that allow remote monitoring and tracking of patients living alone at home or treated in hospitals, and creates a continuum among these through cloud access [4,8].

It is no longer sufficient enough to design just standalone wearable devices, instead it becomes vital to create a complete ecosystem in which sensors in a body area network seamlessly synchronize data to cloud services through the IoT infrastructure [9–11]. The architectural elements generally needed in healthcare IoT systems (Health-IoT) are illustrated in Fig. 1. The architecture includes three main components: (i) body area sensor network, (ii) Internet-connected gateways, and (iii) cloud and big data support. Various applications provide services to different stakeholders in the system through this platform. Data generated from

* Corresponding author at: Department of Computer Science, University of California Irvine, USA.
E-mail address: amirr1@uci.edu (A.M. Rahmani).

http://dx.doi.org/10.1016/j.future.2017.02.014
0167-739X/

Fig. 1. General IoT-based health monitoring system.

sensors attached to users is made available to caregivers, family members and authorized parties giving them the ability to check the subject's vital signs from anywhere at any time.

According to predictions, the current hospital-centered healthcare systems will evolve first to hospital–home-balanced in 2020, and then ultimately to home-centered in 2030 [12]. In order to realize such evolution, new system architectures, technologies, and computing paradigms are required, particularly in the smart spaces and e-Health domains. It should be noted that the paradigm shift towards smart ubiquitous healthcare systems results in new challenges to manifest themselves in fulfilling different system requirements such as reliability, interoperability, energy-efficiency, low-latency response, mobility, security, etc.

Gateways generally act as a hub between a sensor layer and cloud services. With an in-depth observation of a gateway's role in a smart home/hospital, where the mobility and location of users and things are confined to the hospital premises or the building, it can be noticed that the stationary nature of gateways empowers them with the luxury of being non-resource constrained in terms of processing power, power consumption and communication bandwidth. Such a valuable characteristics can be exploited by reinforcing the gateways with sufficient processing power, intelligence, and orchestrated networking capabilities, thus becoming a smart e-Health gateway.

However, the advantageous services that can be potentially offered by a smart gateway will be limited if the gateway is deployed in a standalone and independent fashion. Scalability and mobility issues can easily arise and the efficacy of the solution will be significantly limited. This reveals the demand for an intermediary layer of computation where a geo-distributed network of smart gateways provides intelligence at the edge of the network and facilitates the interplay between sensors layer and cloud layer. This paradigm, which is also called fog or edge computing [13–15], enables the system to support seamless mobility, load balancing, efficient scalability, low-latency response, and developing applications utilizing services offered by multiple sensors and gateways, just to mention a few.

In this paper, which is a major extension of our recent works published in [16], we present a fog computing-based solution to enhance different characteristics of IoT architectures used for healthcare applications in terms of energy-efficiency, performance, reliability, interoperability, to name a few. The main contributions of this article are as follows:

- Presenting a practical solution to take advantage of fog computing in IoT-health systems.
- Elaborating the features of a fog computing based health-IoT system and its services from different perspectives.
- Proposing fog-based mobility support to enable seamless connectivity for mobile sensors.
- Providing a proof of concept full-system implementation from development of cloud services to hardware–software demonstration of our prototype of Smart e-Health Gateways.
- Demonstrating the system with a medical case study called Early Warning Scores (EWS) with hierarchical fog-assisted cloud computing.

The rest of the paper is organized as follows: In Section 2, related work and motivation of this paper are presented. Section 3 describes the architecture of a fog-assisted IoT based e-Health platform using Smart e-Health Gateways. The properties and features of the networked smart e-Health gateways are presented in more detail in Section 4. Demonstration of our Smart e-Health Gateways on a medical case study along with experimental results are provided and discussed in Section 5. Finally, Section 6 concludes the paper.

2. Related work and motivation

In the healthcare context, designing an efficient IoT-based system is a challenging task due to the following main issues. First, the chosen sensor networking technology must be resource-efficient and customized for e-Health applications. Medical sensor nodes, especially implanted ones, have much lower processing power, memory, transmission speed, and energy supply than sensors in other sensor networks domain. Second, unlike common sensor networks where interval-based data transmission is used (e.g., temperature and humidity monitoring), e-Health applications often need to manage streaming-based transmissions where realtime requirements need to be considered. Consequently a considerable energy is dissipated during the transmission process. For instance, Electrocardiogram (ECG) signal transmission requires 4 kbps bandwidth per channel. Third, in multi-patient applications such as in smart hospitals, hardware platforms with a high processing power and parallel processing features (e.g., multi-core processors) are needed in the gateway due to concurrent nature of the workloads. However, as we discuss in this section, the existing general-purpose gateways are not designed for such scenarios. Fourth, reliability in e-Health application is of utmost importance and even short system unavailability often cannot be tolerated. Thus, as we discuss in the following, the limited resources of medical sensor nodes render the use of general purpose gateways inefficient in most circumstances with respect to delay, energy, and reliability.

Using a three tier architecture, with varying computational capacity, for IoT applications is common in both industry and research. The focal point of most of the related works is the gateway used in the middle tier of the IoT architecture. One of many such efforts is presented in [17,18], which proposes gateways to transparently connect sensor networks with different protocols such as ZigBee, Bluetooth, and Ethernet to the Internet. However,

these gateways have limited flexibility as they cannot be customized for different applications. In a different category of related work, Mueller et al. [19] present a gateway called SwissGate which handles and optimize the operation of a sensor network. They specifically apply SwissGate on home automation applications such as measuring heating, ventilation, and air conditioning control (HVAC) parameters. Bimschas et al. [20] aim at providing some levels of intelligence to gateways by enabling them to execute application code. They propose a middleware for the gateway to offer four possible services: protocol conversion, request caching, intelligent caching, and discovery.

Another work by Jong-Wan et al. [21] present a sensor network system comprising of a main server and several sensing-servers acting as gateways and connecting with different sensor networks. Using network-dependent sensing-servers as gateways results in high implementation and hardware cost as well as poor scalability, making such a design inefficient for many IoT applications. In a related work presented in [22], a plug-configurable-play service-oriented generic gateway is proposed in order to provide simple and rapid deployment of various external sensor network applications. The gateway offers a proper level of interoperability by facilitating heterogeneous sensor networks to work together. However, the middleware presented in their work lacks intelligence and runs on PC, limiting its applicability for many IoT applications. In a similar attempt, Guoqiang et al. [23] propose a general purpose smart gateway. It provides pluggable architecture which enables communication among different protocols, unified external interfaces fitting for flexible software development, and flexible protocol to translate different sensor data.

In order to save energy and reduce the cost of smart home, Bian et al. [24] present a new type of intelligent home control system, using an Android Phone as a temporary home gateway instead of the default home gateway. The aim of the work is to automatically shut down the unused devices by predicting user behavior. In a different application domain, the work presented in [25] proposes a prototype of a smart 6LoWPAN (IPv6 over Low power Wireless Personal Area Networks) border router which makes local decisions of health states using a Hidden Markov Model.

In another work, Satyanarayanan et al. [26] propose that mobile devices can use complex algorithms such as facial recognition and language translation to augment human cognition. However, limitations in processing power and long WAN latency are unacceptable. The authors then propose cloudlets which consist of a computer with wireless connectivity in the vicinity of the mobile device. A base virtual machine (VM) is installed in the cloudlets. VM instances are launched from the cloudlets and configured using an overlay script from the mobile device with desired application. The VMs are designed to migrate from one cloudlet to another to allow ubiquity. The authors claim that applications run faster with this configuration due to lower latencies and higher processing power. However, the authors face challenges regarding long launching times. In [27], Stantchev et al. present the benefits of three-level (i.e., fog based) architecture for a smart healthcare infrastructure from servitization and business point of view. However, they only focus on high level architectural modeling aspects and do not discuss real world implementation and experimental evaluation of the services.

Although efforts of using a gateway in IoT have been greatly expanded in recent years, there are only small improvements towards realizing smart gateways streamlined specifically for the healthcare domain. Most of the presented efforts focus on general purpose gateway designs which affects the provided level of intelligence due to lack of information about the application domain. Some of these efforts limit their level of intelligence only for the sake of plug-and-play ability, or reconfigurability to various domains. Some others just focus on specific domains such as smart home.

Existing contributions using a gateway as an intermediary between sensors and cloud storage, consider a minimal role for the gateway, for example applying simple set of rules. In few cases, a gateway is leveraged for domain specific purposes. However, such platforms fail to satisfy the requirements of other domains. In the healthcare sector, particularly for remote health monitoring, a high level of reliability, availability and robustness is demanded. Moreover, security and privacy issues are of critical importance. The purpose of our smart e-Health gateway is to satisfy these domain-specific requirements by customizing gateways for the healthcare domain and providing intelligence closer to patients.

Our proposal is motivated by the fact that in a smart hospital or in-home healthcare, the gateway is in the unique position between both the BAN/PAN/LAN and the wide area network (WAN). This promising opportunity can be exploited by different means such as collecting health and context information from these networks and providing different services accordingly. By geographically distributing and networking smart e-Health gateways, a smart intermediary layer can be formed to provide smooth and efficient healthcare services without limiting the mobility of patients. In general, the motivations of utilizing a Geo-distributed network of smart e-Health gateways for Health-IoT are manifold. The major ones are:

1. To provide local data processing for real-time notification for medical professionals as described in Sections 4.1, 4.3 and 4.4.
2. To secure the sensitive medical data gathered by sensors and keep the privacy of patients, discussed in Section 4.5.
3. To give interoperability for heterogeneous platforms and communication protocols used in medical sensor networks, elaborated in Section 4.6.
4. To provide mobility of patients across the area of coverage of the Fog layer for hospital and home based care, discussed in Section 4.7.
5. To enable the underlying sensor network to be more efficient in terms of energy and communication bandwidth presented in Section 4.8.

The following section provides the system architecture of the overall Health-IoT system.

3. System architecture and the role of fog computing

The large scale implementation of IoT is expected to introduce billions of additional resource-constrained devices connected to the Internet. The majority of these devices, for example wearable and implantable medical sensors, are not capable of storing data they generate. A straightforward design approach is to transfer this data to a cloud for processing. Given the large number of connected devices, the latency of the connection with the cloud could be significant. Moreover, these devices are power and bandwidth constrained, that make them unfit directly to the cloud architecture. Fog Computing [13] is an essential paradigm shift towards a hierarchical system architecture and a more responsive design. As shown in Fig. 2, Fog is an intermediate computing layer between the cloud and end devices that complements the advantages of cloud computing by providing additional services for the emerging requirements in the field of IoT. This intermediate layer is discussed in different terms in various articles, such as Mobile Edge computing [28], Micro-clouds [29], or simply just Edge Computing [30]. The concept behind our smart e-Health gateway is to provide different services at the edge of the network between the smart objects and the cloud.

Fig. 3 shows a detailed view on how the components of a Health-IoT system can be organized in a distributed manner across the three layers to be used in smart hospitals or home. In such

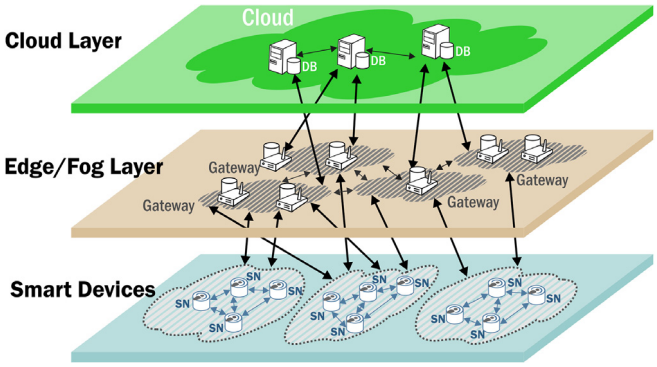

Fig. 2. Generic fog-based IoT architecture.

systems, patient health related information is recorded by body-worn or implanted sensors, with which the patient is equipped for personal monitoring of multiple parameters. This health data can be also supplemented with context information (e.g., date, time, location, temperature). Context-awareness enables to identify unusual patterns and make more precise inferences about the situation. Other sensors and actuators (e.g., medical equipment) can be also connected to the systems to transmit data to medical staff such as high-resolution images (e.g., CAT scan, magnetic resonance imaging). The system architecture includes the following main components:

1. **Medical Sensors and Actuators Network:** Enabled by the ubiquitous identification, sensing, and communication capability, biomedical and context signals are captured from the body and room. The data is then transmitted to the gateway via wireless or wired communication protocols such as Bluetooth, Wi-Fi, ZigBee or 6LoWPAN.
2. **Network of Smart e-Health Gateways:** This layer is built from multiple geographically distributed smart e-Health gateways, i.e., forming the fog. Each gateway, which supports different communication protocols, acts as a dynamic touching point between a sensor network and the local switch/Internet. It receives data from different sub-networks, performs protocol conversion, and provides other higher level services such as data aggregation, filtering and dimensionality reduction.
3. **Back-End System:** Back-end system consists of a cloud computing platform that implements broadcasting, data warehouse and data analytics. Finally, it provides demonstrations for web client as a graphical user interface for final visualization and feedback. The collected health and context data represents a source of big data [31,32] for statistical and epidemiological medical research (e.g., detecting approaching epidemic diseases).

As can be observed from Fig. 3, the intermediate layer is composed of a network of smart e-Health gateways at a strategic location to offer many higher level services to enhance the system characteristics in different aspects. The following section gives an overview of the advantages of this layer in IoT systems.

4. Properties and features of smart e-Health gateways at the fog layer

As mentioned before, the main role of a gateway is to support various wireless protocols and take care of inter-device communication. In this section, we extend its role to become fog enabler by (i) forming an orchestrated network of gateways and (ii) implementing several features such as acting as repository (i.e., local database) to temporarily store sensors' and users' data, and incorporating it with data fusion, aggregation, and interpretation techniques. These are essential to provide local pre-processing of sensors' data, becoming thus a Smart e-Health Gateway.

4.1. Local data processing

As a key feature of fog computing, local data processing is implemented to provide intelligence at the gateway by which the

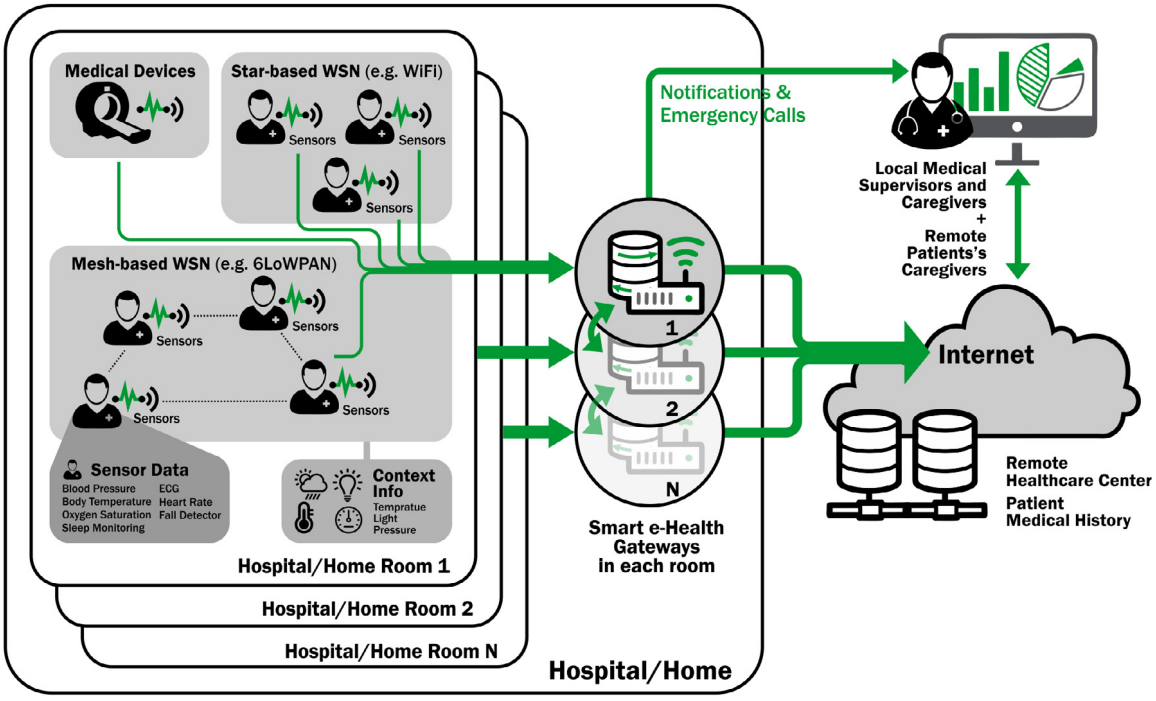

Fig. 3. Components of IoT-based health monitoring system.

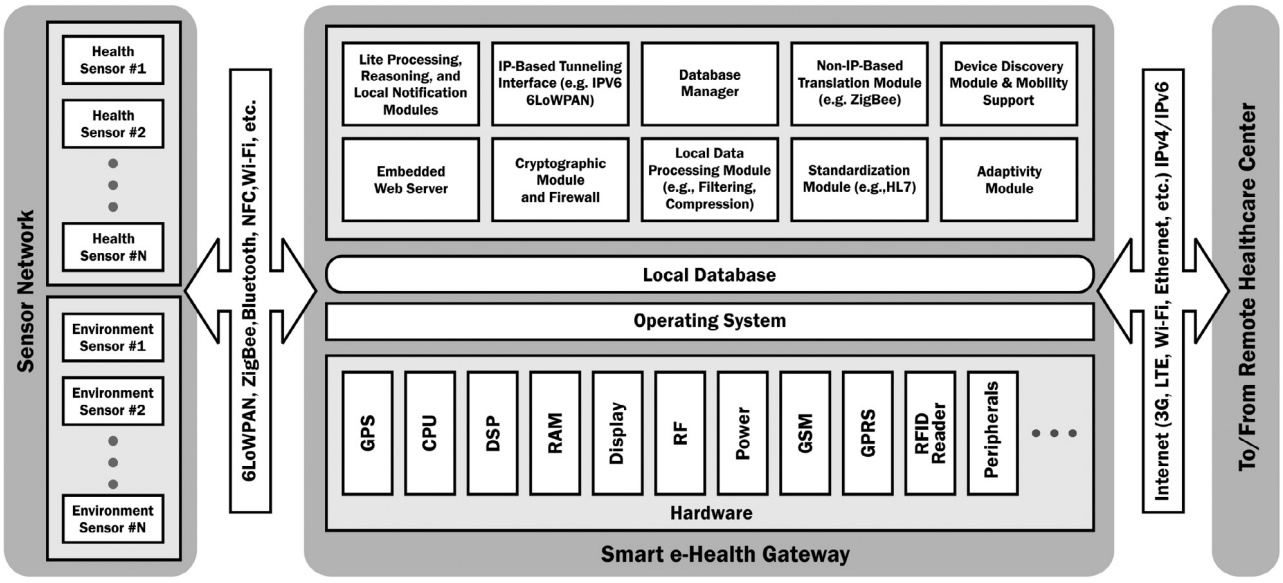

Fig. 4. Smart e-Health gateway architecture.

streaming data is analyzed locally. According to the system architecture, fog layer requires to continuously handle a large amount of sensory data in a short time and response appropriately with respect to various conditions. This task becomes more important in medical cases by enabling the system to react as fast as possible in medical emergencies [33]. Fig. 4 illustrates a conceptual architecture of a smart e-Health gateway utilizing a local processing unit for data filtering, data compression, data fusion, and data analysis.

4.1.1. Data filtering

Receiving data from various sensors makes it essential to implement appropriate pre-processing at the edge before any more advanced processing such as data analysis is performed. Bio-signals (e.g., ECG, EEG and EMG) collected from users' body are the primary sources of information for assessing a patient health status. They usually contain complicated shapes with small amplitude (i.e., in the range of millivolts) and various frequencies. During the sensing of human body, noise is also accumulated to the bio-signals and distort the signal quality. Such noises are produced by different sources such as oscillations of alternating current in the electric power grid, electromagnetic interference from other electrical devices, and improper attachment of sensors to users' body.

The smart e-Health gateway at the fog layer can address this issue as it interfaces sensors directly. The fog layer receives digitalized signals from sensors via various communication protocols. Although sensors may implement light-weight filtering to remove some noises at the data collection phase, more robust and complex data filtering is still required at the fog layer.

4.1.2. Data compression

In the context of data communication, data compression is used for reducing communication latency and energy consumed during transaction. Both lossy and lossless data compression are widely used in the IoT domain depending on the application. Lossless data compression has the shortcoming of requiring rather heavy computation for performing complex algorithms. Therefore, there are processing power requirements in terms of processor speed and memory size when using a lossless data compression method.

In device layer of Health-IoT systems, both lossy and lossless compression methods are useful. However, in many cases lossy data compression is more suitable for resource-constrained sensors due to limitations such as battery life time and available processing power. For instance, many popular lossless ECG compression methods [34–36] do not fit to many types of sensors while lossy compression methods, for example a method introduced by Yu et al. [37], are feasible in terms of hardware requirements. However, for applications such as real-time ECG monitoring, it is desirable to have lossless compression to ensure that all features of the signals all observable with a high precision. Fog computing provides the required computational power for efficiently running complex lossless data compression algorithms by offloading the burden from device layer. Furthermore, it enables real-time operation while using lossless data compression.

4.1.3. Data fusion

Data fusion enables the system to effectively decrease the volume of data, and consequently reduce the energy needed for data transmission. Data fusion is categorized into three classes: complementary, competitive, and cooperative [38]. Complementary data fusion can be performed at the fog layer to achieve better global knowledge. Obtaining temperature difference between body and the environment is an instance provided from two sensory data. Competitive data fusion can be also utilized at the edge in a way that data from a single parameter is collected from different sources to improve the accuracy and consistency of results in case of sensors' failure. Finally, cooperative data fusion can also provide benefits at the edge in a way that new information is extracted in smart gateways from the heterogeneous data collected from diverse sources. For instance, cooperative data fusion can provide comprehensive information about the medical state of a patient from his/her vital signs.

4.1.4. Data analysis

The sensitivity of the system is improved by applying local data analysis at the edge. It can assist the system to detect and predict emergency situations. For instance, in case of fall detection for elderly people, fog layer can locally offer fall-detection related processing rather than sending parameters to a cloud and waiting for the responses. Consequently, the system reacts to the emergency situation faster and more reliable and implements real-time responses. In addition to the sensitivity of the system, utilizing data analysis in the fog layer enables the system to minimize the processing latencies of critical parameters.

Furthermore, local data analysis and local feedback from the sensory data improve the system reliability and consistency in case of unavailability of Internet connection. For long-term remote monitoring of individuals suffering from chronic diseases, Internet disconnection may occur frequently. In this case, fog computing enables to keep the functionality of the system operational locally. Moreover, it is possible to save the sensory data and processing results in a local storage at the fog layer and synchronize them with the Cloud later.

4.2. Adaptivity

Considering the application of fog layer in various cases, some predefined parameters (e.g., transmission rate from sensors to the cloud) are set depending on the use case. However, it is also essential for the fog layer to be reconfigurable and adaptive over time, particularly when critical events take place. The reconfiguration can be dynamically applied for various services utilizing incremental machine learning algorithms such as incremental support vector machine and incremental neural networks.

Data transmission at the fog layer needs to be adaptively tuned. This includes not only data requests from sensors but also data transmission rate from the fog layer to the cloud. For instance, in long-term monitoring of a patient suffering from a cardiovascular disease, the system should learn to increase the request rates (priority) for heart-related parameters when detecting an abnormal heart-related sign. Furthermore, transmission rate to the cloud is performed by assigning priorities to different services and parameters. The priority of data transmission rate to the cloud for patients with acute diseases needs to be higher, while patients suffering from chronic diseases require a lower transmission rate. As a result, adaptive fog computing improves the system performance by increasing sensitivity and specificity of critical parameters.

4.3. Local storage

To ensure that the system can smoothly recover the data, gateways should store the incoming data in a local storage. The operating system on a gateway handles the local repository and stores the data in a non-volatile memory. Based on the type and significance, data can be stored in local storage in a compressed or encrypted way. Data in the repository can be exported to medical standard formats such as Health Level-7 (HL7) [39] if required.

Data storage is also necessary for other functionalities of the gateway. As discussed earlier, gateway is responsible for data analysis, compression, filtering, and encryption, all these functions need a local temporary storage. Because the speed to transfer data from the gateway to the cloud is limited by network bandwidth, and computations are limited by processing power of the gateway, in case of inequality of data processing and data transfer, local storage will act as a cache to implement a continuous data flow. Data storage on gateways makes the system reliable and robust even when network is unavailable. The local repository is handled by the database manager unit shown in Fig. 4.

4.4. Local actuation

In IoT-based healthcare system, actuation can be classified into different forms. It can be used in the form of information streaming, controlling medical actuators, and sensor network reconfiguration. In most of these cases, a predictable and fast response time is demanded. Examples for fast response actuations are adjusting the frequency of electrical nerve stimulation based on the heart rate, or adjusting insulin release rate in automatic pumps based on the patient blood Glucose and other vital signs.

Streaming patient medical signals in real-time to a control panel for medical experts is also a sensitive case to transmission delay where a minimum samples per second rate needs to be met. Using the local processing power and networking facilities, the gateway is able to stream real-time signals such as ECG and PPG (photoplethysmogram) to a client device (i.e., tablet), without relying on the Internet connectivity.

Notifications are also necessary features for smart e-Health gateways at the edge of the network. Health monitoring systems often need to inform and warn medical teams, caregivers, and the patient about an emergency situations. Any failure in the notification service may cause serious problems for both patients and medical treatments. Compared to a cloud server which is able to send notifications via several methods, a gateway has limited resources and can only notify via some specific media. However, the advantage is that gateway-based notifications act independently (e.g., via the local network or GSM) even during unavailability of cloud server, to maximize the reliability of the system and to ensure that users can receive critical notifications in time.

4.5. Security

Security can be considered as one of the most essential requirements in Health-IoT applications on the ground that an unsecured systems can have serious vulnerabilities. In order to provide a high level of security, operating system level techniques can be utilized at gateways such as IPtable offered by Linux. More precisely, IPtables and IPFW provided by Linux kernel firewall can be used for configuring IP packet table which is basically a set of rules for network packets. Typically, IP tables are configured to grant permissions to some ports for communication while other ports are blocked for preventing unnecessary traffic [40]. As the gateway can also act as an embedded web server during network unavailability or whenever needed, it can communicate over secure HTTPS and authenticate sensor nodes to maintain the confidentiality, integrity, and authenticity of the system. Although IPtables provide some advantages, they cannot be considered as a robust tool for security. In order to have a higher level of security, IPtables must be in cooperation with other advanced security methods. To address these issues, different approaches have been recently proposed in the literature [41–43]. However, cryptographic operations in these approaches are heavy in terms of required processing power and energy making them unfeasible for resource-constrained devices. In a recent proposal focused on security, Rahimi et al. [44,45] introduced a secure and efficient authentication and authorization approach for Health-IoT systems which requires some processing power at the edge. More precisely, on the contrary of running secure methods at resource-constrained sensor nodes, the approach exploits properties of a smart gateway in fog computing for heavy and security-related jobs.

4.6. Interoperability and reconfigurability

Besides the standardization efforts, it is evident that interoperability plays a key role to the success of Health-IoT systems. With such heterogeneous mix of networking technologies, protocols and platform choices to implement IoT-based system, integrating these application silos is an evident challenge. Our smart e-Health gateway plays a key role in providing interoperability for the various sensors connected via distinct network interfaces. As shown in Fig. 4, the health sensors and the context sensors are connected to the Smart e-Health Gateway using either wireless network or wired connections while using different standards (e.g., ZigBee, 6LoWPAN, Bluetooth, Wi-Fi) to communicate with the gateway. The smartness of the gateway comes here in the form of easy integration of these heterogeneous networking technologies, protocols

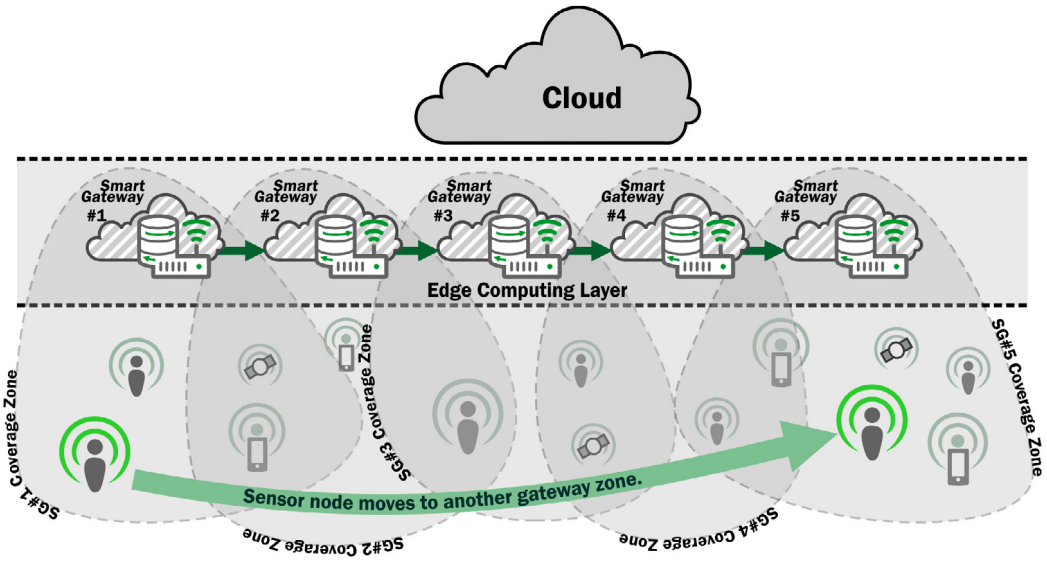

Fig. 5. Node mobility in fog computing.

and standards thereby enabling them to exchange information and work seamlessly.

Technical Interoperability: The various system components in IoT-based system are built by different vendors, and hence they use different network interfaces and standards. In our smart e-Health gateway implementation, technical interoperability is achieved by directing exchanged information to the smart e-Health gateway which has multiple interfaces. Adaptation layers in the gateway facilitate inter protocol exchange of messages and format conversion, which is part of the syntactic interoperability discussed later. One should not have all the possible interfaces in the gateway unless they are used by sensor nodes. This dynamic inclusion or removal of such interfaces is managed by the reconfigurability feature of the smart gateway.

Syntactic Interoperability: Syntactic interoperability layer relies on the previous, technical interoperability layer. It deals with the format of messages exchanged between systems. Once a message is delivered, the receiver has to identify the content of the message and hence the need for the protocol support modules in the central gateway. Variations in protocols result in differences in the format of the message. One common example of tunneling is the case when a 6LoWPAN edge router needs to tunnel between 6LoWPAN and IPv4/IPv6 protocols [46]. Protocol translators can also be used to buffer an incoming message and forward in another format. These functions are realized using the two modules, IP based tunneling interface and non-IP based translation module, shown in Fig. 4.

In addition to network level protocols, medical data is formated in a specific format. Data in any of the Electronic Health Record (EHR) standards, such as HL7 [39], is re-formatted when necessary by the smart gateway, in the standardization module shown in Fig. 4. Sensor nodes can be free from processing overhead that results in formatting the data into standards. In addition, the overhead on the communication channel due to the standards related information that could be sent with the data is removed. This makes the sensor nodes to be energy and bandwidth efficient by sending unformatted data to the gateway.

Semantic Interoperability: To have this common understanding of the meaning of a certain data, a vocabulary (i.e., ontology [47]) of the terms used in that specific context has to be shared first. In addition to the definition of the terms exchanged, the relationship among these terms has to be drawn. Our smart e-Health gateway is designed to provide semantic interoperability in two ways. The first interoperability is for other devices interested in the data collected from sensor nodes. On the other side, the gateway is connected to the Internet and human readable data should be presented.

4.7. Device discovery and mobility support

Mobility in general involves two main processes, handover and roaming, that are needed in order to avoid data loss and service interruptions, and to maintain quality of service (QoS). In a mobile host, handover arises in case of switching from a connection channel to another channel while roaming takes place when moving from one network to another. Mobility can also be categorized into macro and micro types which are defined as mobility between different network domains and within a network domain, respectively [48].

There exists a few methods supporting mobility at edge routers [49]. However, there is no comprehensive method available at the moment to fully address the challenges in the IoT realm. For example, incompatibility with multihop routing, and requirements of NS (Neighbor Solicitation)/NA (Neighbor Advertisement) exchanges are two open issues in the proxy mobile IPv6. NEMO becomes more complicated when different types of mobility (micro and macro) happen simultaneously. There are also some mobility-related mechanisms in the literature handling the mobility support either at the cloud or via additional remote assistant servers. This results in increased handover latency due to often a long distance from a mobile node to the cloud. Particularly in healthcare environments, latency of the network and services need to follow some standards, for example IEEE 1073 [50]. This clearly shows the demand for additional layer between the nodes and the cloud to enhance mobility.

A simplified view of how smart gateways are used in the fog layer to assist nodes during mobility from one geographic location to another domain is shown in Fig. 5. Device discovery helps in identifying a new node entering to the domain under primary control of the associated gateway. Taking a single node that intends to move from gateway #1 to gateway #6, in the path shown by the arrow, each gateway utilizes device discovery and mobility support module to provide uninterrupted service for the node. The initial configuration shows that each gateway manages a set of nodes. In [51], we show in detail how to provide device discovery and

mobility at the edge, in particular for interoperability purposes. As a node moves, it receives a broadcast message from the gateways regarding its identity. When the node receives the broadcast message it replies with a discovery request to the respective gateway, which is processed by the device discovery and mobility support module in the gateway. At the fog layer, the two gateways (the source and destination) exchange information regarding the profile of the node and handle the handover process.

4.8. Energy efficiency for sensor nodes

Data processing at sensor nodes has several drawbacks due to sensor nodes' resource constraints. In some cases, complex algorithms might be successfully executed at sensor nodes, however at a high cost of energy. Therefore, shifting selective signal processing tasks from sensor nodes to smart gateways at the fog layer can be an effective solution to address the aforementioned issues, particularly when gateways are not battery-powered.

Recently, several approaches [52,53] have focused on providing energy efficiency for Health-IoT applications. In [52], Otto et al. perform real-time signal processing on sensor nodes while Gia et al. [53] utilize a low power transmission protocol to save transmission energy for sensor nodes. Although such techniques enhance energy efficiency of sensor nodes, a considerable amount of energy can be still saved via fog computing by outsourcing some loads from sensor nodes to smart gateways.

4.9. Latency

For a continuous 24/7 remote health monitoring system, rapid decision making and agile responses are essential for several acute diseases and emergencies, where data processing and transmission time should be minimized. In cloud computing where raw data is transferred from sensor nodes to cloud, if network condition is unpredictable, it may cause uncertainty to response latencies. The situation is more critical when streaming-based data processing is needed (e.g., signal processing on ECG or EEG signals). Comparatively, implementing high priority data analytics in distributed smart gateways and making critical and time-sensitive decisions within the local network make the system more robust and predictable. The processed data can be then transmitted to the cloud for storage and further analysis. Moreover, in a large scale sensor network, local signal processing at the fog layer can minimize the traffic between gateways and the cloud.

5. Demonstration and evaluation

To demonstrate our hypothesis, *an enhanced healthcare IoT system realized through the use of a network of smart e-Health gateways at the fog layer*, a set of demonstrations and evaluations are presented in this section. It starts with demonstrators and evaluations that show the characteristics and performance of the smart gateway and the benefits they provide. These demonstrations show the behavior of a single gateway in a standalone condition as well as the collaborative benefits which can be provided via a layer of networked smart gateways. A medical case study follows the features of the gateways to demonstrate a medical early warning scenario, where a network of gateways forming a fog layer.

To begin with the demonstrations, we present the architecture of our demonstration system and evaluate the system characteristics including higher-level functions developed on the gateway. In our architecture, the system implementation is divided into three major phases: node implementation, networked smart gateways implementation, and back-end and user interface implementation. Our Smart e-Health Gateway, called UT-GATE, collaborates with sensor nodes, other smart gateways, remote server, and clients. The implemented system architecture is shown in Fig. 6. To demonstrate the device and protocol level interoperability of UT-GATE, we have implemented different network topologies, such as mesh and star topologies, using several wireless sensor technologies, like 6LoWPAN, Wi-Fi and Bluetooth, so that each sensor in each subnetwork utilizes different platform but works in harmony with UT-GATE. The tunneling interface module in UT-GATE is used by the Mesh-based 6LoWPAN network to interoperate with the rest of the system i.e., to tunnel between 6LoWPAN and IPv4/IPv6 protocols. The non-IP based translation module supports the star-based Bluetooth network to interoperate with the IP-based system, i.e., to translate between Bluetooth and IPv4/IPv6 protocols.

As shown in Fig. 7, UT-GATE is constructed from combination of Pandaboard [54] and Texas Instruments (TI) SmartRF06 board integrated with CC2538 module [55] and MOD-ENC28J60 Ethernet Module [56]. The Pandaboard is low-power, low-cost single-board computer development platform based on TI OMAP4430 system-on-chip following OMAP architecture and fabricated using 45 nm technology for providing high-performance [54]. OMAP4430 processor is composed of microprocessor unit (MPU) subsystem including dual-core ARM Cortex-A9 cores with symmetric multiprocessing at up to 1.2 GHz each. Pandaboard can support different operating systems such as Windows CE, WinMobile, Symbian, Linux, and Palm, and integrate on-chip memory and external memory interfaces, and support memory management and connecting peripherals. In our implementation, 8 GB external memory added to the Pandaboard and powered by Ubuntu operating system which allows to control devices and services such as local storage and notification. Furthermore, it supports different network interfaces such as 802.11 b/g/n (based on WiLink 6.0), Bluetooth v2.1 + Enhanced Data Rate (BDR) (based on WiLink 6.0), and Onboard 10/100 Ethernet.

Bluetooth sink node is created by configuring Bluetooth module (WiLink 6.0) while Bluetooth sensor node is constructed by the combination of Bluetooth module, Arduino Due and analog front-end (AFE) devices. E-health data collected from the AFE devices is sent to the Arduino Due through SPI connection after digitization and then the data is transferred to the sink node through the Bluetooth module. All operations in the Arduino Due are executed on FreeRTOS [57], an open source real-time operating system.

A Wi-Fi network is built by the combination of a sink node constructed by configuring the Wi-Fi Module (WiLink 6.0) in the Pandaboard and a Wi-Fi sensor node formed by using an RTX4140 module [58] which includes micro controller unit (EMF32), Wi-Fi module (Atheros) and the AFE devices. Similar to the Bluetooth module, AFE devices are connected to the RTX module through SPI connection. Unlike Bluetooth, a UDP client running on RTXOS [59], an operating system for the RTX module, sends the data to the Wi-Fi sink node.

The SmartRF06 along with the CC2538 module form the sink node for the 6LoWPAN network collect data from other 6LoWPAN nodes and forward to the Pandaboard through the Ethernet port. Wi-Fi is used for data exchange between the Pandaboard and the remote server.

The SmartRF06 board is the motherboard for low-power RF ARM Cortex M3 based SoCs from Texas Instruments [55]. As shown in Fig. 7, the board is plugged to the CC2538 module to collect the data from the 6LoWPAN subnetwork and then send the data to the Pandaboard via the plugged MOD-ENC28J60 Ethernet Module. To enable communication between the 6LoWPAN nodes and UT-GATE, we used RPL (IPv6 Routing Protocol for Low-power and Lossy Networks) implementation in Contiki OS [60] which is an open source operating system focusing on low-power IoT devices.

The sensor nodes receive Electrocardiogram (ECG), Electroencephalogram (EEG), and Electromyogram (EMG) digital signals through 2 SPI connectors from AFE devices (i.e., TI ADS1292 [61]

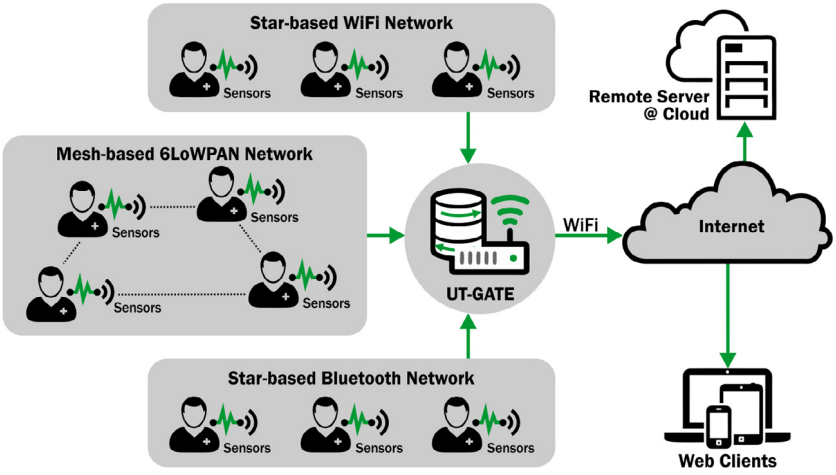

Fig. 6. Architecture overview.

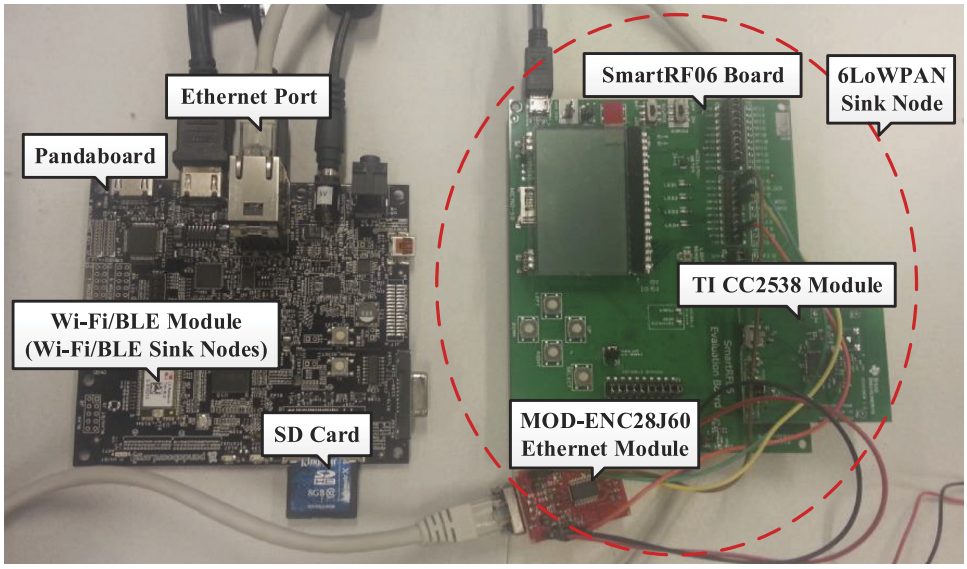

Fig. 7. UT-GATE: Our demonstrator of a smart e-Health gateway.

Table 1
Hardware specification of sensor nodes and UT-GATE.

Device	Micro-controller	Flash (KB)	RAM (KB)	EEPROM (KB)	Clock (MHz)	Voltage (V)
Zigduino R2	ATMega 128	128	16	4	16	3.3
Arduino Uno R3	ATMega 328P	32	2	1	16	5
Arduino Mega	ATMega 1280	128	8	4	16	5
Arduino Due	ARM CortexM3	512	86	–	84	3.3
Zolertia Z1	MSP430	92	8	–	16	3.3
TI-CC2538	ARM Cortex M3	Up to 512	32	–	32	3.3
Pandaboard	Dual-core ARM Cortex-A9	Up to 32 000	1000	–	1200	5

Table 2
Power consumption of sensor nodes when transmitting at 8.7 kbps.

Communication type	Current (mA)	Voltage (V)	Power consumption (mW)
6LoWPAN node	24.6	3.3	81.2
Wi-Fi node	114	3.3	376.2
Bluetooth 2.0 node	56.9	3.3	187.7
BLE node	31.6	3.3	104.4

and TI ADS1298 [62]). The analog front-end devices get analog values from electrodes and perform analog to digital conversion. In SPI connection between a node and ADS1292, a node (i.e., CC2538 module) acts as the master while the ADS129x module act as the slave. The specification and power consumption of the sensor nodes are respectively shown in Tables 1 and 2.

In our IoT-based health monitoring system, the remote server is responsible for handling client requests by providing the requested

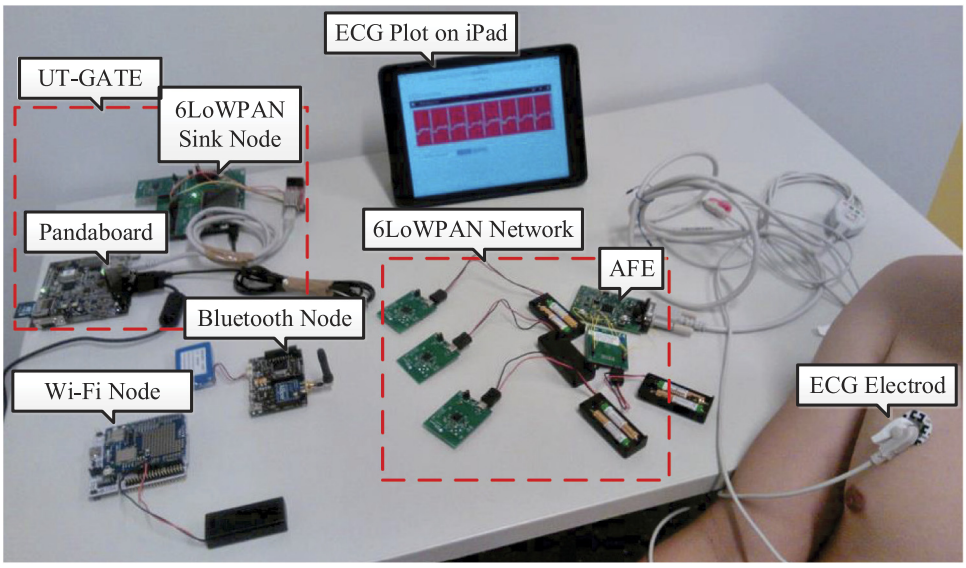

Fig. 8. IoT-based health monitoring system demonstration.

Table 3
Sensing to actuation latency comparison for local fog based vs. remote cloud based scenarios.

Latency of the sensing-to-actuation loop	using Wi-Fi (ms)	using BLE (ms)
Fog-based (locally via UT-GATE)	21	33
Cloud-based (remotely via the Cloud)	161	176

data along with graphical user interface. For the remote server over the cloud implementation, we used the free service provided by "heliohost.org" including MySQL server with remote access facility. The database along with tables are created in the server and the server side scripts are developed in PHP.

As shown in Fig. 8, we have implemented an IoT-based health monitoring system including 6LoWPAN, Wi-Fi and Bluetooth sub-networks to practically demonstrate the features of UT-GATE. The currently implemented functionalities of UT-GATE such as data compression, data fusion, WebSocket server and local storage are discussed in detail in the following subsections.

Table 3 shows benefits of the local fog-based decision making compared to that of the centralized cloud computing scenario in terms of latency. As can be seen from the table, the latency for the sense–decide–act loop for the local scenario is considerably lower then its counterparts making it a viable option for real-time streaming or actuation purposes.

5.1. Compression at UT-GATE

Applying data compression into remote e-Health monitoring systems helps reducing the size of data transmitted over a network. The amount of compressed data varies depending on a particular data compression method. Some methods can achieve a high compression ratio, for example 8:1, 9:1 or even higher, while others cannot reach to these ratios. The key requirement when applying data compression in real-time monitoring systems is computation time of the compression and decompression process because maximum latency for ECG, EMG, EEG signals have to be less than 500ms, according to IEEE 1073 [63]. With the purpose of fulfilling this latency requirement, a LZW [64] algorithm is implemented. LZW is a lossless data compression method which provides rapid compression and decompression. In addition, balanced execution time for compression and decompression in this method helps keeping harmony between input and output. In order to provide a general view of data compression in a real-time e-Health monitoring system, data compression is applied at both sensor nodes and gateways in distinct processes. Even though these compression algorithms are among the least expensive methods available in the literature in terms of minimum hardware requirements, it is still not feasible nor efficient to implement them at sensor nodes. Therefore, data compression is operated at the fog layer as a unit in the smart e-Health gateway.

The LZW algorithm reacts differently to various data sizes. It is more efficient with a higher input size in terms of computation time. For example, when the size of input data increases 10 times, computation time increases about 8 times. However, the computation cost in terms of latency rapidly grows by increasing the data size. Therefore, the data size needs to be carefully chosen to meet the real-time requirements of e-Health data.

Table 4 shows the time required for compression and decompression and compressed, and data size with the number of connected sensor nodes to UT-GATE varying from 1 to 50. Data is collected from sensor nodes having 8 channels for EMG with sampling rate of 1000 samples/second. In addition, other signals including environmental data is obtained resulting to a sample size of approximately 70B. When 120 samples are used as an input, the compression takes around 3.1ms. Transmission time for sending 70B data using 6LoWPAN technology from a sensor node to a gateway, is around 40 μs when transmission rate is 250 kbps. This shows that the transmission time is negligible compared to the compression/decompression time, and total required time for compression, transmission, and decompression is tolerable for real-time transmission of e-Health data. It can be also observed that the smart gateway at the fog has the ability to serve a large number of sensor nodes with still reasonable compression and decompression time.

5.2. Benefits of data processing at UT-GATE

In an IoT-based health monitoring system, as presented in Fig. 3, there are options to apply signal processing at each layer of the

Table 4
Compression results at UT-GATE and latency reduction.

Number of senors nodes connected to UT-GATE	1	2	5	10	50
Number of analog channels	8	8	8	8	8
Data size (120 samples) (B)	8400	16 800	42 000	84 000	420 000
Compressed data size (B)	808	1597	3893	7696	38333
Compression time (ms)	3.1	4.4	9.2	16.6	73.0
Decompression time (ms)	3.3	4.6	11.3	23.0	83.9
Total time of comp., tran., and decomp. (ms)	12.86	21.77	51.64	101.16	463.5
Transmission time of non-processed data (ms)	67.2	134.4	336	672	3360
Total latency reduction (%)	80.8	83.8	84.6	84.8	86.1

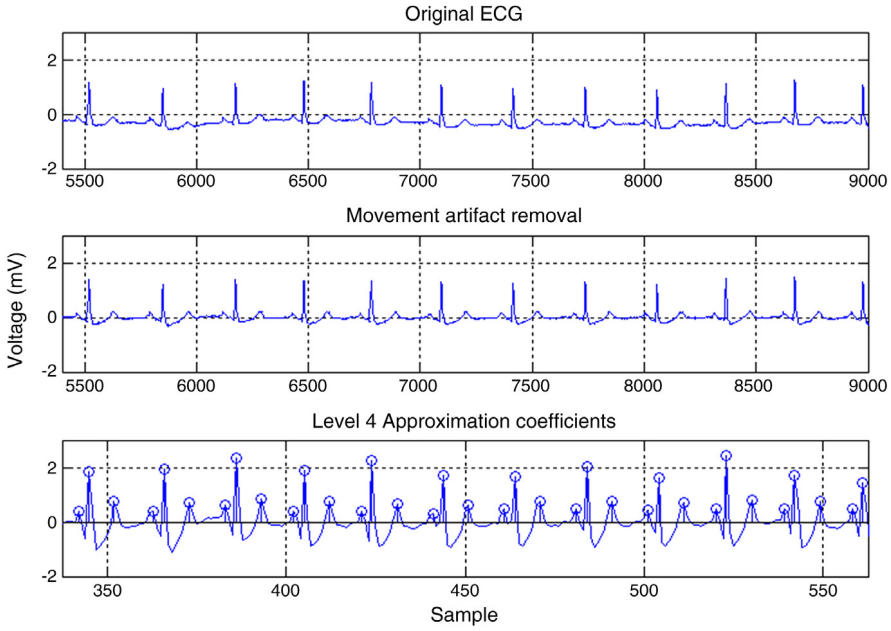

Fig. 9. ECG processing implementation.

system architecture. As introduced in Sections 4.8 and 4.9, fog layer has adequate computational resource and can bring benefits to both sensor node energy efficiency and data transmission time in the system.

As a widely used function in health monitoring applications, we chose to implement ECG signal processing for movement artifact removal and feature extraction (i.e., heart rate calculation from *R* to *R* interval, *P* wave, and *T* wave). We implement this function under three different scenarios: (i) on a sensor node, where the processed data and extracted features are sent to the cloud through UT-GATE, (ii) on a UT-GATE where the raw data is received from sensor nodes and the results are sent to a cloud (fog-assisted cloud computing), and (iii) on the cloud, where raw data is passed from two prior layers. For all these scenarios, we measured energy consumption of sensor node, the size of samples transmitting from UT-GATE to cloud, and the latency of data delivery between them.

We use MIT-BIT Arrhythmia database [65] which includes abundant ECG data sources sampled at 360 samples per second in 11-bit over 10 mV. Data is pre-stored in sensor node or UT-GATE before being processed. The ECG threshold-based feature extraction algorithm is applied after moving average filter and four level discrete wavelet transformation with Daubechies 4 wavelet. Discrete wavelet transformation is a time-frequency method which is suitable for non-stationary signal processing. With each level of transformation, the signal is divided into approximation coefficients and detail coefficients in even data sizes which represent low and high part of the signal respectively. ECG features lie in the low frequency part, so approximation coefficients are kept in each level for the next level transformation. ECG signal processing and peak detection are shown in Fig. 9.

To access the overheads of local processing on sensor nodes in terms of energy consumption, application execution time and energy consumption are measured from Arduino Due acting as a sensor node. Sensor node in real-time application needs to accumulate data for a few seconds before processing. In our case, the collection time for 1000 samples is 2.78 s. A low-cost Wi-Fi module ESP8266 is utilized to transmit data from the sensor node to the UT-GATE. The energy consumption calculation and comparison for the above mentioned three scenarios are presented in Table 5. As can be seen from the table, by outsourcing the processing burden from the sensor node to the smart gateway, about 55.7% of energy is saved thanks to fog computing.

As shown in Table 6, fog computing considerably reduces the data transmission to the cloud due to local data processing. For every 1000 raw samples, it provides 74.1% reduction in sample size, due to sample downsizing through wavelet decomposition in gateway. Meanwhile, essential information about ECG, including waveform and extracted features, are kept and sent to cloud, instead of transmitting raw samples. Consequently, data transmission time from gateway to cloud is reduced, especially for large data sets. The latency reduction when transmitting 240 KB raw samples of data between gateways and the cloud under different Wi-Fi network conditions are presented in Table 7. It can be observed from the table that fog computing can reduce the communication traffic in particular in high-traffic network conditions.

5.3. Local storage, notification, and security at UT-GATE

The UDP server running at the gateway on port 5700 receives data modules for the 6LoWPAN network and under RTXOS on

Table 5
Providing energy efficiency for sensor nodes.

	Time	Current	Energy
Processing at sensor node (collection + execution)	2.78 s + 101 ms	10.1 mA + 106.8 mA	127.34 mJ
Process at UT-GATE/cloud (transmitting raw data)	95 ms	180 mA	56.34 mJ

Table 6
Number of transmitted samples from fog to cloud.

	Processing at sensor node or UT-GATE	Processing at cloud	Improvement (%)
No. of samples	259	1000	74.1

Table 7
Latency reduction between gateways and cloud server for 240 KB of raw samples.

Network condition	Data rate (Mbits/s)	Raw samples trans. time (ms)	Raw samples proc. time + trans. time of processed samples (ms)	Latency reduction (%)
Light load	18	106.6	96.3 + 6.6	3.5
Medium load	12	152.2	96.3 + 9.5	30.5
Heavy load	9	213.3	96.3 + 13.5	48.5

Table 8
XML status code and description.

Code	Description
0	Invalid request or Error
1	Notification – {total in number}
2	No new notification

RTX Wi-Fi modules for the Wi-Fi network. Similarly, the Bluetooth nodes send data to the Bluetooth module in UT-GATE. Received data is processed and stored in the local repository apart from forwarding the same data to the remote server. We have implemented a local repository on UT-GATE using MySQL database which offers several engines. Federated engine is one such kind of engine used to create references to the tables in the remote server without the requirement of database mirroring or replication. The tables created with federated engine on local repository will hold of the same structure and record as in the remote server. Whenever new rule is added or existing rule is updated in remote server, the same information is available in the local repository which helps to give least priority to synchronization process on rules. During processing, if the received data does not conform to the rules, a notification will be logged in the repository. The notification table will be populated directly from local repository and the notification mechanism configured in the remote server will act accordingly.

The gateway purges the locally stored information in repository, which is received 30 min earlier ensuring that data synchronization with the remote server has been successfully completed. If the connection with the remote server is not available, it will store the data as long as possible and begin to delete old data to accommodate new data, if it runs out of memory. During network unavailability, UT-GATE can also act as a local web server by handling the client application's request along with notifications taking its operation to a higher level. While acting as local web server, it will send responses either in XML or JSON format as requested, leaving the user interface rendering at the client-end by utilizing the client resources efficiently to minimize its resource usage.

Notification mechanism configured in the gateway can be used on permanent basis parallel to remote server or whenever the connection to the remote server is unavailable. For notification implementation, we have developed an Android application which communicates with UT-GATE over Wi-Fi on demand. If new notification exists, the gateway communicates with the respective node with a message along with node ID and timestamp in XML format.

The XML format has two sections: header and content. The header section contains the status of the current request in the form of code and description. Currently, gateway responds with 3 status codes; the codes along with its description are given in Table 8. When mobile application receives response, at first it will check the code in status header and if status returns '1', it will proceed with content section, otherwise it will deliver the status header description as message.

As mentioned before, UT-GATE has been powered by the Ubuntu operating system which comes with a firewall called Uncomplicated Firewall (UFW) which is used to restrict the accessibility of protocols and ports. With proper configuration, the gateway can be tuned to achieve certain degree of security level. For our implementation, we blocked all ports and protocols except TCP and UDP over ports 80, 443, 3306 and 5700.

5.4. WebSocket server on UT-GATE

An embedded WebSocket server was implemented on UT-GATE using the Tornado non-blocking Web server framework for Python. The server receives data as a UDP server directly from the sensor nodes functioning as a UDP client. Another configuration involves receiving the signal from the MySQL database configured to serve as a streaming database. The benefit of this approach is multi-user support for the WebSocket server since the signal is always stored and can be retrieved many times. On the client side, a WebSocket enabled browser accesses an HTML page hosted at the gateway that offers the JavaScript interface and necessary parameters to establish the two-way asynchronous WebSocket link between the browser and the gateway. The ECG signal is buffered to 400 samples and sent as WebSocket messages of 800 bytes each averaging a data rate of 1.1 KB/s. In our LAN setup, it takes 32 ms for the Web client to receive the packet and render the continuous chart. The buffer size can be decreased to lower the latency at the expense of higher processing overhead for the Web Client. A JavaScript client plots the near real-time chart and a set of commands is implemented to control transmission start-stop. Future work includes the expansion of the command set into a complete API for gateway management and a generic library capable of listening to different transport layer protocol sockets for easy interoperability of variety of nodes with different protocols.

5.5. Medical early warning scores: A case study

Early Warning System is a guide tool in hospitals for estimating the degree of illness and predicting the risk of deterioration to reduce the complications and prevent intensive care unit admission. It is based on recording patient's medical parameters periodically to find abnormal signs. This guide system works based on the fact

Table 9
A typical early warning score model [70].

Physiological parameters	3	2	1	0	1	2	3
Respiration rate (breaths/min)	≤8		9-10	12-20		21-24	≥25
Oxygen saturation (%)	≤ 91	92-93	94-95	≥96			
Temperature (°C)	≤35.0		35.1-36.0	36.1-38.0	38.1-39.0	≥39.1	
Systolic BP (mmHg)	≤90	91-100	101-110	111-219			≥220
Heart rate (beats/min)	≤40		41-50	51-90	91-110	111-130	≥131
Level of consciousness				A*			V,P or U*

that clinical deterioration is a visible pattern in patient's vital sign up to 24 h before deterioration happens [66,67,68]. The system is designed based on a method called Early Warning Score (EWS) which calculates a total score using a set of values related to some medical parameter. This score reflects the health status of a patient with respect to her/his vital signs. To find the score, a nurse should measure and record the vital signs of a patient in an observation chart. The nurse marks a vital sign with a score, based on its value in its range. A higher score means more abnormalities of a specific vital sign and the sum of all scores indicates the overall health status of a patient [69]. Table 9 shows a typical EWS model.

Considering a patient's final calculated score, medical staff can modify the therapy orders and recording intervals. Such a scoring method for medical early warning was presented for first time in 1997 [71]. Initial versions of the EWS were using five vital signs: heart pulse, respiration rate, blood pressure, body temperature, and blood Oxygen level (SpO2). Several years after first implementations of this method in hospitals, some enhancements were proposed and applied to the original EWS algorithm. Modified early warning system (MEWS) [72], standardized early warning system (SEWS) [73], and a national early warning system in UK (NEWS) [74] are some examples. The main difference of these enhanced methods is the number of parameters they use to calculate the final score. Also they customize the threshold values of vital signs based on the average property value in the target country.

Despite many benefits offered by EWS in hospitals in terms of reduced mortality rate and healthcare costs, some researchers have reported false diagnosis and errors mainly caused by inaccurate records. The necessity of implementing an accurate early warning system drives hospitals to move towards automatic electronic solutions [75].

The objective of this case study is to demonstrate an in-home early warning system using the discussed system architecture and the concepts of fog computing. Therefore, we implemented an in-home EWS system where patients are supposed to be provided with medically acceptable services and environments. To this end, it is essential to monitor environmental properties as well as patient's activities to consider their effect on patient's vital signs. The components in our proposed system comprises three layers shown in Fig. 10.

At the first layer, we implemented a network of sensors which are divided into three groups. The first and main group includes medical sensors to monitor vital signs such as heart rate, respiration rate, body temperature, blood pressure, blood oxygen level, and ECG. The second group of sensors contains environmental sensors for recording light, temperature, and humidity of the room. Activity sensors are in the last group for recording the patterns of movements, posture, and total daily steps of the patient.

At the second layer, data received from sensor network is handled by UT-GATE via wireless communication. The gateway receives data from several type of medical sensors via a UDP server implemented by Node.js. This UDP server stores data for each sensor in separate files considering the information of patient whom the data is coming from. Another service running on UT-GATE with Apache server reads data from locally created files for further processings using a Python script. As properties of the collected data are not similar, an adaptation is needed to unify the data structure. The frequency of data collection is not the same for all the sensors, ranging from 250 samples per second (e.g., ECG) to about 5 samples per day (e.g., step counts). Communication protocols also differs for different sensors (e.g., Bluetooth and Wi-Fi with UDP and TCP transmission protocols).

Node.js UDP server is responsible for handling different data transmission protocols while continuously receiving data from sensor network. In the gateway, using Python, preliminary data analysis is implemented to detect critical conditions before performing detailed analysis at the cloud. UT-GATE first filters the data for noise reduction. A bandpass filter (0.5–100 Hz) with Finite Impulse Response (FIR) is implemented in the gateway to reduce noises from the incoming ECG signal (250 samples per second). Then, heart rate data is extracted from the signal considering RR intervals. Fig. 11 demonstrates a window of raw and filtered ECG data.

Moreover, the gateway applies data fusion on the sensory data. In this regard, two heart rate signals from the monitored person are collected using two different devices. Due to the presence of inevitable noises (e.g., movement's noises) on the acquired signals, two obtained data may not be identical during the monitoring. To reduce the noise impact, first, outliers and meaningless data are removed using an anomaly detection method. Considering a threshold for the minimum value of heart rate, zero values are removed. In our case, outliers in heart rate data sources are mainly because of improper probe connections, often due to loose connection or low conductivity between the probe and the patient's skin (low moisture). Second, a weighted average is implemented for two sources values to achieve more reliable heart rate (see Fig. 12). The weight of each data set is determined by the sensor accuracy mentioned in the sensor's datasheet.

After the aforementioned steps, real-time EWS calculation, based on the constraints and rules shown in Table 9, are performed at the fog layer. We believe EWS is a proper case study for fog computing in healthcare, as it demands reliability, rapid response, and real-time processing, and deals with heterogeneous sensory data which calls for pre-processing. To calculate the EWS score, a window of collected vital signs are chosen. Using a rule-based system defined based on Table 9, the score is calculated at UT-GATE. In our use case, a 35 years old male subject (BMI = 28.3) is monitored for 8 h. Fig. 13 shows one minute of monitoring in which the subject encountered a sudden variation in his vital signs. Fig. 14 also demonstrates the calculated score. The increasing score indicates the higher probability of the patient's health deterioration for this specific window. Therefore, in this case, the medical experts are notified about the critical state of the patient for further considerations.

In parallel with the real-time analysis and notification, the gateway applies compression and encryption to the filtered and processed data, sends feedback and notifications to the sensor network and cloud respectively, and stores the data in UT-GATE's local storage (see Fig. 10).

We compress processed data in the gateway to have a backup for the situation when Internet connectivity drops off. The compression method we used here is different with the method described earlier for the real-time data compression during transmissions. In this case study, we use *tar* method to create a file

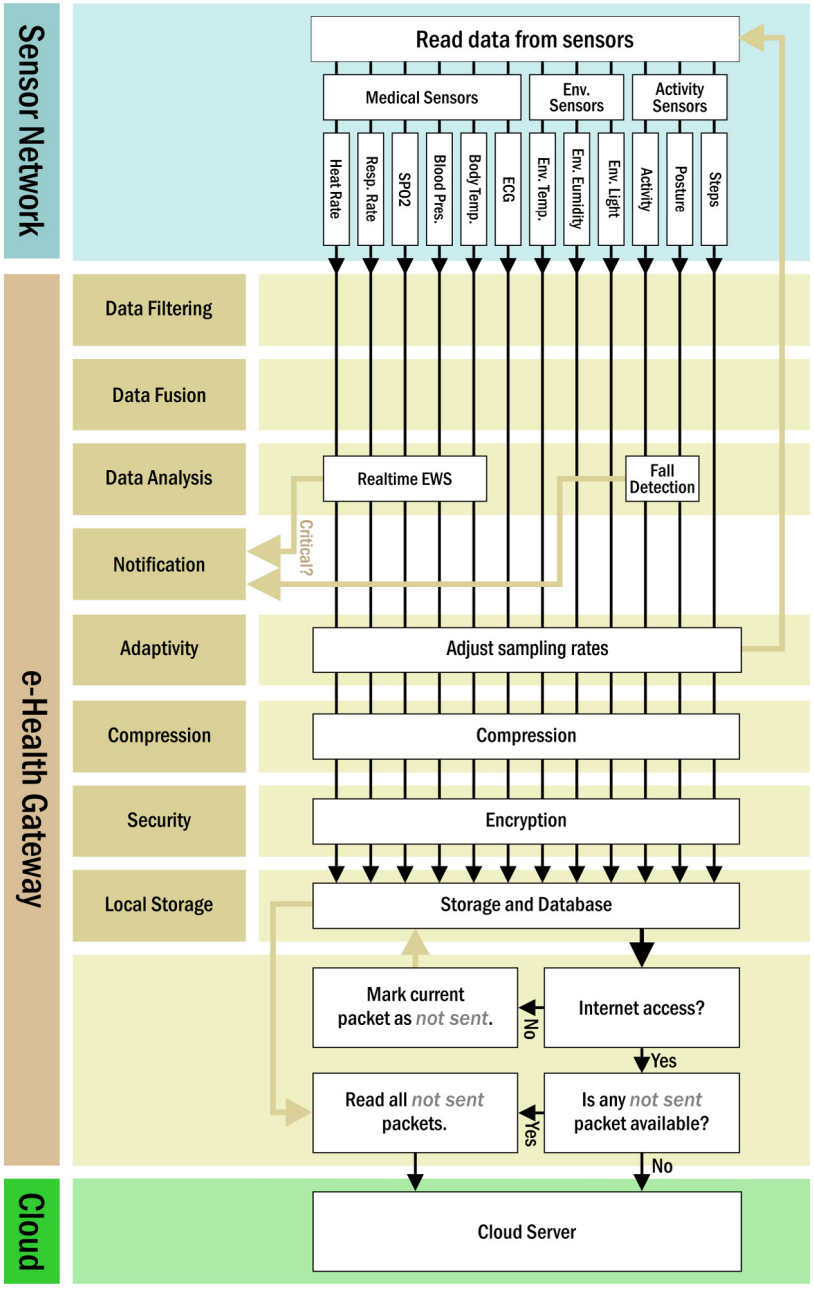

Fig. 10. Fog-based EWS System services, components, and data flow.

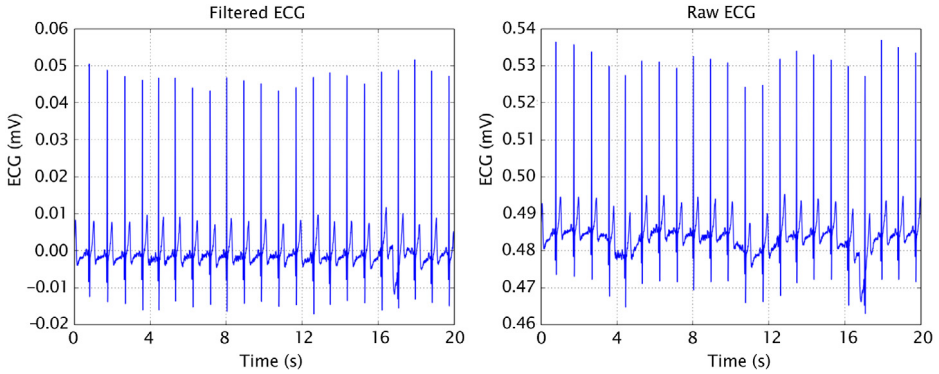

Fig. 11. Raw and filtered ECG data.

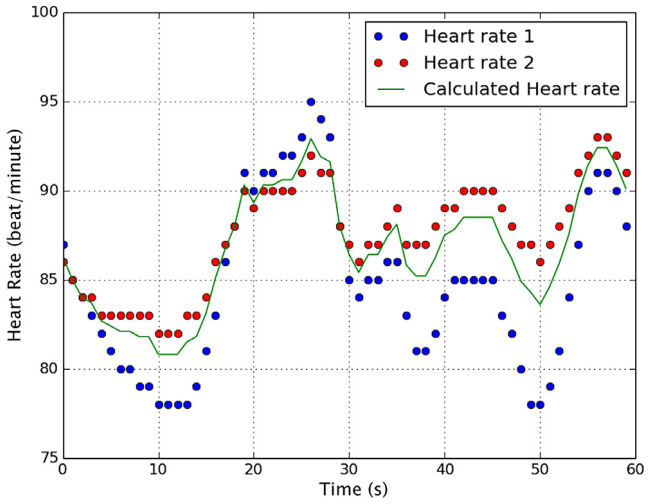

Fig. 12. Heart rate data fusion.

as temporary location for collected data and use *tar.gz* method to compress temporary file when the size of file reaches to a certain value (e.g., 500 KBytes). The effect of temporary file size on the compression ratio is shown in Fig. 15. The larger the size of temporary file, the higher the compression ratio would be. However, the compression ratio shows an insignificant improvement for temporary files larger than 500 KByte.

To keep stored data secure, we use an asymmetric encryption method using *Crypto* library [76] in Python. All the compressed files on the gateway are encrypted with a public key and only the data collector service in the cloud has the private key to decrypt the files.

The gateway adjusts the sampling rate according to the calculated EWS score. When the score increases, the gateway considers priority for patient's health situation by increasing the sensor's sampling rate to track the momentary changes more accurately. On the other hand, when the score is zero or within the normal range, the sampling rate is reduced by sending feedback to sensor network in order to improve the energy efficiency.

Local storage consists of a file storage and database to keep the properties and indexes of the files. In the last phase, the gateway checks the availability of the Internet connection and sends the data to the could server. In case of connectivity issues, it tags the unsent data to be sent in future. A process in local storage service checks and synchronizes the stored data with the cloud server and removes old and duplicated files from the storage, and deletes their indexes from the database.

The third layer of our in-home early warning system consists of a cloud server and user interface for patients, caregivers, and

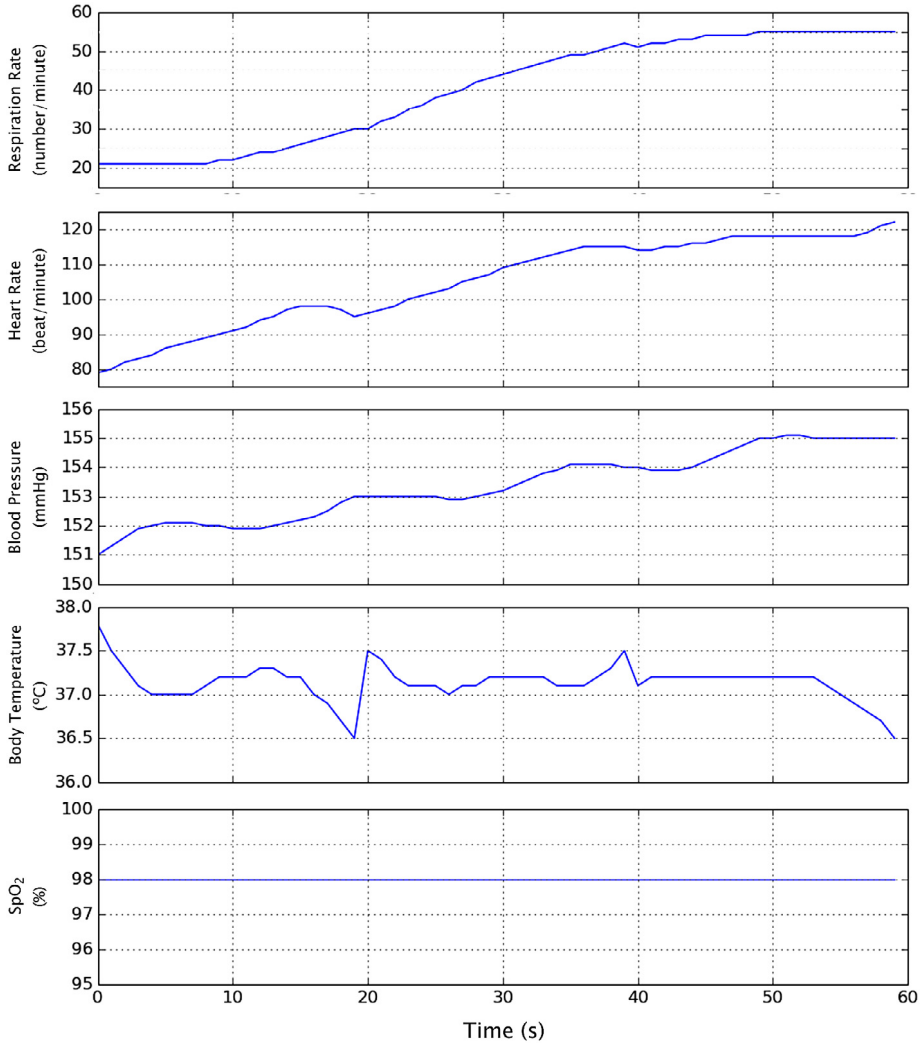

Fig. 13. Vital signs.

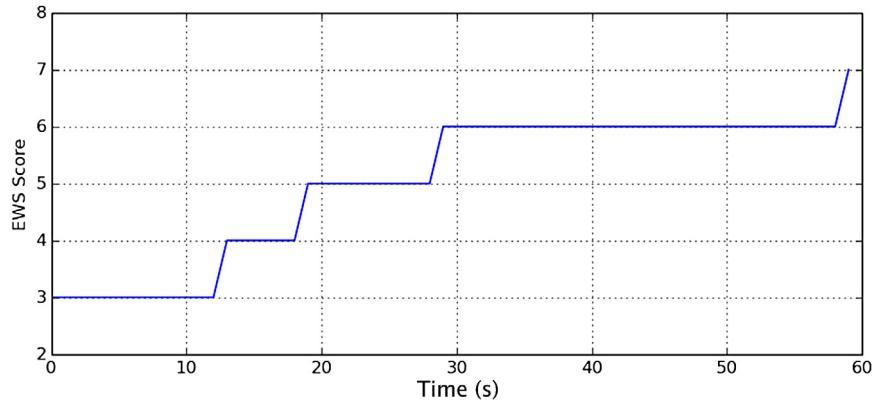

Fig. 14. Calculated EWS score.

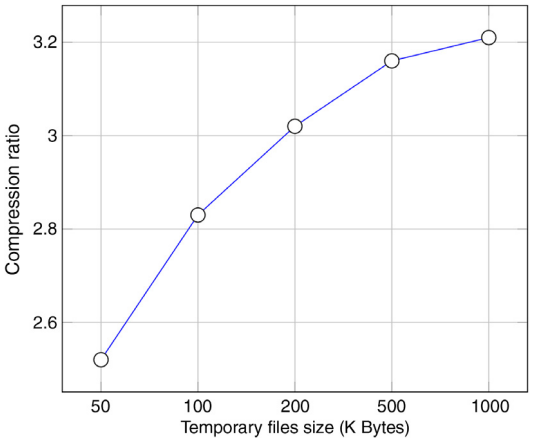

Fig. 15. The compression ratio for different temporary file size.

medical experts. At this layer, the cloud server receives and records gateway's data (e.g., EWS score, vital signs and notifications) for further processing. Using the incoming data from the gateway and the patient medical history data, the server provides reports, suggestions and possible alerts for medical experts and caregivers. Cloud server also has a user interface to provide data access for health professionals. Fig. 16 shows a snapshot of the developed HTML5 web-based user interface which is used as a control panel. In addition, patients and caregivers have also access to an Android-based smartphone application to receive notes and notifications.

6. Conclusions

In this paper, the concept of fog computing and Smart e-Health Gateways in the context of Internet-of-Things based healthcare systems was presented. Smart gateways at the close proximity of sensor nodes in smart home or hospital premises can exploit their unique strategic position to tackle many challenges in IoT-based health systems such as mobility, energy efficiency, scalability, interoperability, and reliability issues. We investigated in detail a range of high level services which can be offered by smart gateways to sensors and end-users in a Geo-distributed fashion at the edge of the network (e.g., local processing, storage, notification, standardization, firewall, web services, compression, etc.). We presented a proof of concept implementation of an IoT-based remote health monitoring system which includes our demonstration of a Smart e-Health Gateway called UT-GATE. By exploiting a number of UT-GATEs, we formed an intermediary processing layer to demonstrate the fog computing concept for IoT-based healthcare systems. Our fog-assisted system was applied to a medical case study called Early Warning Scores, targeted to monitoring patients with acute illnesses. Our full system demonstration includes all the data flow processes from data acquisition at sensor nodes to the cloud and end-users.

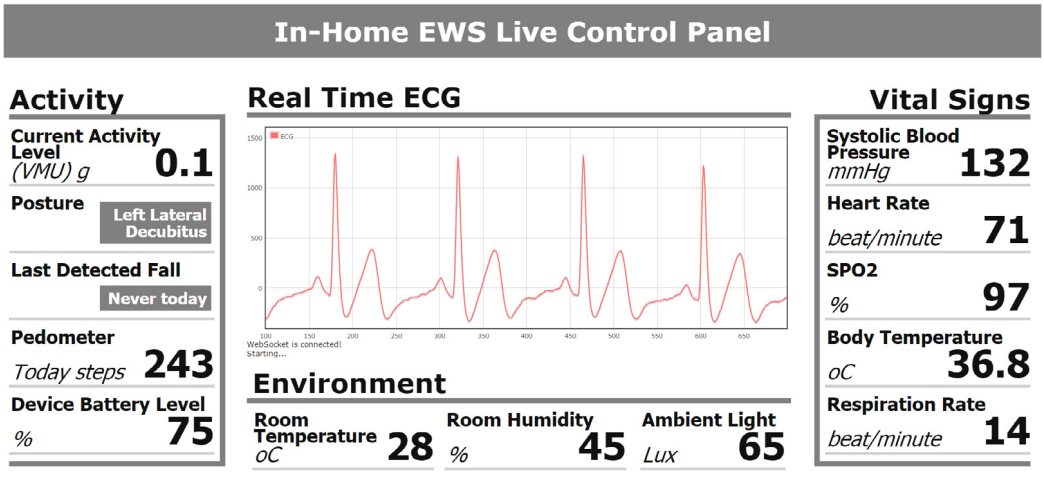

Fig. 16. Live control panel web interface.

References

[1] European Commission Information Society, Internet of things strategic research roadmap, 2009. http://www.internet-of-things-research.eu/. (Accessed 14 July 2015).

[2] European Commission Information Society, Internet of things in 2020: a roadmap for the future, 2008. http://www.iot-visitthefuture.eu. (Accessed 14 July 2015).

[3] A. Dohr, R. Modre-Opsrian, M. Drobics, D. Hayn, G. Schreier, The internet of things for ambient assisted living, in: Proceedings of the International Conference on Information Technology: New Generations, 2010, pp. 804–809.

[4] D. Miorandi, S. Sicari, F. De Pellegrini, I. Chlamtac, Internet of things: Vision, applications and research challenges, Ad Hoc Netw. 10 (7) (2012) 1497–1516.

[5] M. Carmen Domingo, An overview of the internet of things for people with disabilities, J. Netw. Comput. Appl. 35 (2) (2012) 584–596.

[6] Hairong Yan, Li Da Xu, Zhuming Bi, Zhibo Pang, Jie Zhang, Yong Chen, An emerging technology â wearable wireless sensor networks with applications in human health condition monitoring, J. Manag. Anal. 2 (2) (2015) 121–137.

[7] Y.J. Fan, Y.H. Yin, L.D. Xu, Y. Zeng, F. Wu, IoT-based smart rehabilitation system, IEEE Trans. Ind. Inf. 10 (2) (2014) 1568–1577.

[8] B. Farahani, F. Firouzi, V. Chang, M. Badaroglu, K. Mankodiya, Towards fog-driven IoT eHealth: Promises and challenges of IoT in medicine and healthcare, Elsevier Future Generation Computer Systems (2017).

[9] European Research Cluster on the Internet of Things, IoT semantic interoperability: research challenges, best practices, solutions and next steps, IERC AC4 Manifesto - "Present and Future", 2014.

[10] B. Xu, L.D. Xu, H. Cai, C. Xie, J. Hu, F. Bu, Ubiquitous data accessing method in IoT-based information system for emergency medical services, IEEE Trans. Ind. Inf. 10 (2) (2014) 1578–1586.

[11] L. Jiang, L.D. Xu, H. Cai, Z. Jiang, F. Bu, B. Xu, An IoT-oriented data storage framework in cloud computing platform, IEEE Trans. Ind. Inf. 10 (2) (2014) 1443–1451.

[12] C.E. Koop, R. Mosher, L. Kun, J. Geiling, E. Grigg, S. Long, C. Macedonia, R. Merrell, R. Satava, J. Rosen, Future delivery of health care: Cybercare, IEEE Eng. Med. Biol. Mag. 27 (6) (2008) 29–38.

[13] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog computing and its role in the internet of things, in: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, 2012, pp. 13–16.

[14] M. Aazam, E.N. Huh, Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT, in: 2015 IEEE 29th International Conference on Advanced Information Networking and Applications, 2015, pp. 687–694.

[15] M. Aazam, E.N. Huh, Fog computing and smart gateway based communication for cloud of things, in: Future Internet of Things and Cloud (FiCloud), 2014 International Conference on, 2014, pp. 464–470.

[16] A.-M. Rahmani, N.K. Thanigaivelan, Tuan Nguyen Gia, J. Granados, B. Negash, P. Liljeberg, H. Tenhunen, Smart e-Health gateway: Bringing intelligence to IoT-based ubiquitous healthcare systems, in: Proceeding of 12th Annual IEEE Consumer Communications and Networking Conference, 2015, pp. 826–834.

[17] K.A Emara, M. Abdeen, M. Hashem, A gateway-based framework for transparent interconnection between WSN and IP network, in: Proceedings of the EUROCON, 2009, pp. 1775–1780.

[18] Q. Zhu, R. Wang, Q. Chen, Y. Liu, W. Qin, IoT gateway: Bridging wireless sensor networks into internet of things, in: Proceedings of the International Conference on Embedded and Ubiquitous Computing, 2010, pp. 347–352.

[19] R. Mueller, J.S. Rellermeyer, M. Duller, G. Alonso, Demo: A generic platform for sensor network applications, in: Proceedings of the IEEE International Conference on Mobile Adhoc and Sensor Systems, 2007, pp. 1–3.

[20] D. Bimschas, H. Hellbrück, R. Mietz, D. Pfisterer, K. Römer, T. Teubler, Middleware for smart gateways connecting sensornets to the internet, in: Proceedings of the International Workshop on Middleware Tools, Services and Run-Time Support for Sensor Networks, 2010, pp. 8–14.

[21] J.-W. Yoon, Y.-K. Ku, C.-S. Nam, D.-R. Shin, Sensor network middleware for distributed and heterogeneous environments, in: Proceedings of the International Conference on New Trends in Information and Service Science, 2009, pp. 979–982.

[22] L. Wu, Y. Xu, C. Xu, F. Wang, Plug-configure-play service-oriented gateway - for fast and easy sensor network application development, in: Proceedings of the International Conference on Sensor Networks, 2013, pp. 53–58.

[23] S. Guoqiang, C. Yanming, Z. Chao, Z. Yanxu, Design and implementation of a smart IoT gateway, in: Proceedings of the International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, 2013, pp. 720–723.

[24] J. Bian, D. Fan, J. Zhang, The new intelligent home control system based on the dynamic and intelligent gateway, in: Proceedings of the International Conference on Broadband Network and Multimedia Technology, 2011, pp. 526–530.

[25] W. Shen, Y. Xu, D. Xie, T. Zhang, A. Johansson, Smart border routers for eHealthCare wireless sensor networks, in: Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing, 2011, pp. 1–4.

[26] M. Satyanarayanan, P. Bahl, R. Caceres, N. Davies, The case for VM-based cloudlets in mobile computing, IEEE Perv. Comput. 8 (4) (2009) 14–23.

[27] Vladimir Stantchev, Ahmed Barnawi, Sarfaraz Ghulam, Johannes Schubert, Gerrit Tamm, Smart items, fog and cloud computing as enablers of servitization in healthcare, Sensors Transducers 185 (2) (2015) 121.

[28] S. Nunna, A. Kousaridas, M. Ibrahim, M. Dillinger, C. Thuemmler, H. Feussner, A. Schneider, Enabling real-time context-aware collaboration through 5G and mobile edge computing, in: Proceedings of 12th International Conference on Information Technology - New Generations, 2015, pp. 601–605.

[29] Shiqiang Wang, R. Urgaonkar, Ting He, M. Zafer, K. Chan, K.K. Leung, Mobility-Induced service migration in mobile micro-clouds, in: Proceedings of Military Communications Conference, 2014, pp. 835–840.

[30] Yi Lin, B. Kemme, M. Patino-Martinez, R. Jimenez-Peris, Enhancing edge computing with database replication, in: Proceedings of 26th IEEE International Symposium on Reliable Distributed Systems, 2007, pp. 45–54.

[31] D. Laney, 3D Data Management: Controlling Data Volume, Velocity, and Variety, Technical Report, Application Delivery Strategies by META Group Inc., 2001.

[32] M. Beyer, Gartner Says Solving 'Big Data' Challenge Involves More Than Just Managing Volumes of Data, 2016. http://www.gartner.com/newsroom/id/1731916.

[33] J. Granados, A.M. Rahmani, P. Nikander, P. Liljeberg, H. Tenhunen, Towards energy-efficient HealthCare: An internet-of-things architecture using intelligent gateways, in: Proc. of International Conference on Wireless Mobile Communication and Healthcare, 2014, pp. 279–282.

[34] M.L. Hilton, Wavelet and wavelet packet compression of electrocardiograms, IEEE Trans. Biomed. 44 (5) (1997) 394–402.

[35] Z. Lu, D. Youn Kim, W.A. Pearlman, Wavelet compression of ECG signals by the set partitioning in hierarchical trees algorithm, IEEE Trans. Biomed. 47 (7) (2000) 849–856.

[36] R. Benzid, A. Messaoudi, A. Boussaad, Constrained ECG compression algorithm using the block-based discrete cosine transform, Digit. Signal Process. 18 (1) (2008) 56–64.

[37] F. Touati, R. Tabish, U-healthcare system: State-of-the-art review and challenges, J. Med. Syst. 37 (3) (2013).

[38] H.F. Durrant-Whyte, Sensor models and multisensor integration, Int. J. Robot. Res. 7 (6) (1988) 97–113.

[39] Health Level Seven Int'l. Introduction to HL7 Standards, 2012. www.hl7.org/implement/standards. (Accessed 30 July 2015).

[40] netfilter/iptables - nftables project, http://netfilter.org/projects/nftables/. (Accessed 24 July 2015).

[41] G. Kambourakiset, E. Klaoudatou, S. Gritzalis, Securing medical sensor environments: The codeblue framework case, in: Proceeding of the Second International Conference on Availability, Reliability and Security, 2007, pp. 637–643.

[42] R. Chakravorty, A programmable service architecture for mobile medical care, in: Proceeding of Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, 2006, pp. 1–5.

[43] J. Ko, J.H. Lim, Y. Chen, R. Musvaloiu-E, A. Terzis, G.M. Masson, T. Gao, W. Destler, L. Selavo, R.P. Dutton, MEDiSN: medical emergency detection in sensor networks, ACM Trans. Embed. Comput. Syst. 10 (1) (2010) 11:1–11:29.

[44] S.R. Moosavi, T.N. Gia, A. Rahmani, E. Nigussie, S. Virtanen, H. Tenhunen, J. Isoaho, SEA: A secure and efficient authentication and authorization architecture for IoT-based healthcare using smart gateways, in: Proceeding of 6th International Conference on Ambient Systems, Networks and Technologies, 2015, pp. 452–459.

[45] S.R. Moosavi, T.N. Gia, E. Nigussie, A. Rahmani, S. Virtanen, H. Tenhunen, J. Isoaho, Session resumption-based end-to-end security for healthcare internet-of-things, in: Proceeding of IEEE International Conference on Computer and Information Technology, 2015, pp. 581–588.

[46] Z. Shelby, C. Bormann, 6LoWPAN: The Wireless Embedded Internet, Wiley Publishing, 2010.

[47] T. Bittner, M. Donnelly, S. Winter, Ontology and semantic interoperability, in: Proceeding of Large-Scale 3D Data Integration Challenges and Opportunities, 2005.

[48] Z. Shelby, C. Bormann, 6LoWPAN: The Wireless Embedded Internet, Wiley, UK, 2009.

[49] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, B. Patil, Proxy Mobile 2 IPv6, Internet Engineering Task Force, 2008.

[50] IEEE Standard for Medical Device Communication, Overview and Framework, 1996.

[51] B. Negash, A.M. Rahmani, T. Westerlund, P. Liljeberg, H. Tenhunen, LISA: Lightweight internet of things service bus architecture, Proc. Comput. Sci. 52 (2015) 436–443. The 6th International Conference on Ambient Systems,

Networks and Technologies, ANT-2015, the 5th International Conference on Sustainable Energy Information Technology, SEIT-2015.
[52] C. Otto, A. Milenković, C. Sanders, E. Jovanov, System architecture of a wireless body area sensor network for ubiquitous health monitoring, J. Mob. Multimedia 1 (4) (2006) 307–326.
[53] T. Nguyen Gia, N.K Thanigaivelan, A.M. Rahmani, T. Westerlund, P. Liljeberg, H. Tenhunen, Customizing 6LoWPAN networks towards Internet-of-Things based ubiquitous healthcare systems, in: Proceeding of NORCHIP, 2014, pp. 1–6.
[54] OMAP® 4 PandaBoard System Reference Manual, 2010. http://pandaboard.org. (Accessed 04 August 2015).
[55] SmartRF06 Evaluation Board User's Guide, 2013. http://www.ti.com/lit/ug/swru321a/swru321a.pdf. (Accessed 04 August 2015).
[56] Olimex, MOD-ENC28J60 Development Board, Users Manual, 2008. https://www.olimex.com/Products/Modules/Ethernet/MOD-ENC28J60/resources/MOD-ENC28J60.pdf. (Accessed 04 August 2015).
[57] R. Barry, Using the FreeRTOS Real Time Kernel, Microchip PIC32 Edition, FreeRTOS Tutorial Books, 2010.
[58] RTX41xx Low Power Modules, http://www.rtx.dk/RTX41xx_Modules-4024.aspx. (Accessed 04 August 2015).
[59] RTX Real-Time Operating System, http://www.keil.com/rl-arm/kernel.asp. (Accessed 04 August 2015).
[60] A. Dunkels, B. Gronvall, T. Voigt, Contiki - a lightweight and flexible operating system for tiny networked sensors, in: Proceeding of International Conference on Local Computer Networks, 2004, pp. 455–462.
[61] Texas Instruments, Low-Power, 2-Channel, 24-Bit Analog Front-End for Biopotential Measurements, 2012.
[62] Texas Instruments, ECG Implementation on the TMS320C5515 DSP Medical Development Kit (MDK) with the ADS1298 ECG-FE, 2011.
[63] IEEE standard for medical device communication, overview and framework, in: ISO/IEEE 11073 Committee, 1996.
[64] V.S. Miller, Data compression method, US4814746 A, Filing date Aug 11, 1986, Publication date Mar 21, 1989.
[65] G.B. Moody, R.G. Mark, The impact of the MIT-BIH arrhythmia database, IEEE Eng. Med. Biol. Mag. 20 (3) (2001) 45–50.
[66] J. McGaughey, F. Alderdice, R. Fowler, A. Kapila, A. Mayhew, M. Moutray, Outreach and Early Warning Systems (EWS) for the prevention of Intensive Care admission and death of critically ill adult patients on general hospital wards, Cochrane Database Syst. Rev. (3) (2007).
[67] C. Franklin, J. Mathew, Developing strategies to prevent inhospital cardiac arrest: analyzing responses of physicians and nurses in the hours before the event, Crit. Care Med. 22 (2) (1994) 244–247.
[68] R.M. Schein, N. Hazday, M. Pena, B.H. Ruben, C.L. Sprung, Clinical antecedents to in-hospital cardiopulmonary arrest, Chest 98 (6) (1990) 1388–1392.
[69] M. Odell, C. Victor, D. Oliver, Nurses' role in detecting deterioration in ward patients: systematic literature review, J. Adv. Nurs. 65 (10) (2009).
[70] D. Georgaka, M. Mparmparousi, M. Vitos, Early warning systems, Hosp. Chronicle 7 (1) (2012) 37–43.
[71] R.J.M. Morgan, F. Williams, M. Wright, An early warning scoring system for detecting developing critical illness, Clin. Intensive Care 8 (2) (1997) 100.
[72] R. Paterson, et al., Prediction of in-hospital mortality and length of stay using an early warning scoring system: clinical audit, Clin. Med. 6 (3) (2006) 281–284.
[73] G.D. Barlow, Standardised early warning scoring system, Clin. Med. 6 (4) (2006).
[74] Royal College of Physicians, National Early Warning Score (NEWS): standardising the assessment of acute-illness severity in the NHS, 2012.
[75] A. Zarabzadeh, M. O'Connell, J. O'Donoghue, T. O'Kane, S. Woodworth, J. Gallagher, S. O'Connor, F. Adam, Features of electronic early warning systems which impact clinical decision making, in: Proceeding of 25th International Symposium on Computer-Based Medical Systems, 2012, pp. 1–4.
[76] PyCrypto API Documentation, https://pythonhosted.org/pycrypto/. (Accessed 26 May 2016).

Amir M. Rahmani received his Master's degree from Department of Electrical and Computer Engineering, University of Tehran, Iran, in 2009 and Ph.D. degree from Department of Information Technology, University of Turku, Finland, in 2012. He also received his MBA jointly from Turku School of Economics and European Institute of Innovation & Technology (EIT) ICT Labs, in 2014. He is currently an EU Marie Curie Global Fellow at University of California Irvine, USA, and TU Wien, Austria. He is also adjunct professor in embedded parallel and distributed computing at University of Turku, Finland. He is the author of more than 120 peer-reviewed publications. His research interests span Self-aware Computing, Energy-efficient Many-core Systems, Runtime Resource Management, Healthcare Internet of Things, and Fog Computing.

Tuan Nguyen Gia: received his B.Sc. (Tech.) degree in Information technology from Department of Information Technology, Helsinki Metropolia University of Applied Sciences, Helsinki, Finland in 2012, and M.Sc. (Tech.) degree in Information Technology, Embedded Computing from the Department of Information Technology and Communication Systems, University of Turku, Finland in 2014. He is currently working towards his Ph.D. degree at the University of Turku, Finland. His research interests include Internet of Things (IoT), Smart Healthcare, and Medical cyber–physical system, FPGA and Wireless Body Sensor Networks.

Behailu Negash received B.Sc. degree in Electrical Engineering from Mekelle University (Ethiopia) and M.Sc. (Tech.) degree in Embedded Computing from University of Turku (Finland) in 2006 and 2015, respectively. He is currently a Ph.D. student in Embedded Electronics laboratory, IoT4Health research group, at University of Turku. His research focuses on architecture and interoperability of Internet of Things, network architecture and embedded software.

Arman Anzanpour is Ph.D. student in "IoT for Health" group at University of Turku since September, 2014. He received his Master in Biomedical Engineering from Amirkabir University of Technology, Iran and his Bachelor in Material Engineering from Ferdwosi University of Mashhad. His current research focuses on Internet of Things and smart health monitoring frameworks.

Iman Azimi received his bachelor degree in Biomedical Engineering at University of Isfahan (Iran) in 2010, and his master degree in Artificial Intelligence and Robotics at Sapienza, University of Rome (Italy) in 2014. He started his Ph.D. research in IoT4health group, department of Information Technology at University of Turku in August 2015. His current research area is intelligent health big data analytics based on Internet-of-Things.

Mingzhe Jiang is currently a Ph.D. student in IoT4Health research group in University of Turku. She received her M.Sc and B.Sc degree in Instrument Science and Technology from Harbin Institute of Technology in the year 2014 and 2012. Her current research is related with bio-signal processing and pattern recognition in Healthcare Internet of Things.

Pasi Liljeberg received the M.Sc. and Ph.D. degrees in Electronics and Information Technology from the University of Turku, Turku, Finland, in 1999 and 2005, respectively. He is a Senior University Lecturer in Embedded Electronics Laboratory and an adjunct professor in embedded computing architectures at the University of Turku, Embedded Computer Systems laboratory. During the period 2007–2009, he held an Academy of Finland researcher position. He is the author of more than 200 peer-reviewed publications, has supervised nine Ph.D. theses. Liljeberg is the leader of the Internet-of-Things for Healthcare (IoT4Health) research group.