Energy-Efficient Location and Activity-Aware On-Demand Mobile Distributed Sensing Platform for Sensing as a Service in IoT Clouds

Charith Perera, *Member, IEEE*, Dumidu S. Talagala, *Member, IEEE*, Chi Harold Liu, *Senior Member, IEEE*, and Julio C. Estrella, *Member, IEEE*

***Abstract*—The Internet of Things (IoT) envisions billions of sensors deployed around us and connected to the Internet, where the mobile crowd sensing technologies are widely used to collect data in different contexts of the IoT paradigm. Due to the popularity of Big Data technologies, processing and storing large volumes of data have become easier than ever. However, large-scale data management tasks still require significant amounts of resources that can be expensive regardless of whether they are purchased or rented (e.g., pay-as-you-go infrastructure). Further, not everyone is interested in such large-scale data collection and analysis. More importantly, not everyone has the financial and computational resources to deal with such large volumes of data. Therefore, a timely need exists for a cloud-integrated mobile crowd sensing platform that is capable of capturing sensors data, on-demand, based on conditions enforced by the data consumers. In this paper, we propose a context-aware, specifically, location and activity-aware mobile sensing platform called context-aware mobile sensor data engine (C-MOSDEN) for the IoT domain. We evaluated the proposed platform using three real-world scenarios that highlight the importance of *selective sensing*. The computational effectiveness and efficiency of the proposed platform are investigated and are used to highlight the advantages of context-aware selective sensing.**

***Index Terms*—Activity awareness, cloud-sensing middlware platforms, context awareness, data filtering, distributed sensing, Internet of Things (IoT), location awareness, selective sensing.**

I. Introduction

THE INTERNET of Things (IoT) [1] has become popular over the past decade. As part of the IoT infrastructure, sensors are expected to be deployed all around us, from everyday objects we use, to public infrastructure such as bridges and roads [2], [3]. As the price of sensors diminishes rapidly, we can soon expect to see very large numbers of objects comprising sensors and actuators. In addition, the modern technology-savvy world is already full of devices comprising sensors, actuators, and data processors. The concentration of computational resources will enable the sensing, capturing, collection, and processing of real-time data from billions of connected devices, and can be envisaged to serve many different applications including environmental monitoring, industrial applications, business, and human-centric pervasive applications [4].

IoT allows people and things to be connected any time, any place, with anything and anyone, ideally using any path/network and any service [5]. IoT is expected to generate large volumes of sensor data [4]. Due to the latest innovations in the computer hardware sector and the reduction in hardware costs, large-scale data processing is becoming increasingly economical. Specially, with the popularity of utility-based cloud computing [6] that offers computational resources in a "*pay-as-you-go*" model, the tendency to collect a large amount of data has been increasing over the last few years. In 2010, the total amount of data on earth exceeded one zettabyte (ZB). By the end of 2011, the number grew up to 1.8 ZB [4]. Further, it is expected that this number will reach 35 ZB in 2020. It is therefore apparent that sensor data have significant value if we can collect and extract insights from them.

Along with the IoT concepts, business models such as sensing as a service have also generated significant interest [7]. The sensing as a service model envisions a marketplace where sensor data are traded in an open and transparent manner with interested consumers. Sensing as a service can therefore be seen as a platform where data owners can sell data to interested sensor date consumers in "pay-as-you-go" fashion. On the one hand, such a model stimulates the growth of sensor deployments. On the other hand, it reduces the cost of sensor data acquisition due to its shared nature (i.e., sense once, sell to many). In addition, the sensing as a service model will also share the common IoT infrastructure to collect, process, and store data. In contrast, crowd sensing technologies have been widely used to collect sensor data in IoT paradigm. In community sensing, also referred to as group sensing [8] and mobile *crowdsensing* [9], the focus has been on monitoring large-scale phenomena that cannot be measured using information from a single individual. The purpose here is to collect information from a large group of people in order to analyze and use that information for the benefit of the group as a whole.

Manuscript received June 05, 2015; revised December 26, 2015; accepted January 03, 2016. Date of publication February 03, 2016; date of current version February 25, 2016.

C. Perera is with the Department of Computing, Faculty of Maths, Computing, and Technology, Open University, Milton Keynes MK7 6AA, U.K. (e-mail: charith.perera@open.ac.uk).

D. S. Talagala is with the Centre for Vision, Speech, and Signal Processing, University of Surrey, Guildford GU2 7XH, U.K. (e-mail: d.talagala@surrey.ac.uk).

C. H. Liu is with The Beijing Institute of Technology, Beijing 100081, China, and also with Sejong University, Seoul, South Korea (e-mail: chiliu@bit.edu.cn).

J. C. Estrella is with the Institute of Mathematics and Computer Science (ICMC), University of São Paulo, São Paulo 13566-590, Brazil (e-mail: jcezar@icmc.usp.br).

Digital Object Identifier 10.1109/TCSS.2016.2515844

2329-924X © 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

In the discussion so far, we briefly introduced the IoT, sensing as a service model, and the Big Data in the IoT paradigm. In this paper, we define nonselective sensing as the process of collecting sensor data from all possible sensors available, all the time without any filtering. While we acknowledge the importance and value of collecting large volumes of sensors data, a number of drawbacks of nonselective sensor data collection exist. Despite the fact that nonselective data collection could generate more value in the long term (e.g., due to discovery of knowledge that were not intended during the time of data collection), it definitely creates a problem (or difficulties) in the short term. The main issue in nonselective data collection is cost. Moreover, the processing and storing of data lead to more costs directly associated to the computational resource requirements (e.g., CPU, memory, and storage space). Further, processing more data requires more time which creates the problem of not being able to extract knowledge from the collected data on time. Crucially, another issue is energy consumption. Sensors are typically resource-constrained devices with limited access to energy. Nonselective sensing therefore leads to significant energy consumption and faster battery drain which create additional challenges related to the IoT infrastructure maintenance. Another challenge is network communication. Large-scale data transfers over the network without any kind of filtering lead to the continuous use of the communication radios continuously. This also leads to faster battery drain in addition to the heavy network traffic generated in the IoT infrastructure. Thus, energy is a critical factor, especially in the crowd sensing domain, where humans are involved in maintaining the sensing infrastructure. Therefore, we believe that on-demand selective sensing (i.e., perform sensing only under certain conditions) enables to avoid all the issues discussed above. To this end, we propose a scalable energy-efficient data analytics platform for on-demand distributed mobile crowd sensing called C-MOSDEN.[1]

This paper is organized as follows. In Section II, we define the problem domain in detail. The functional requirements of the proposed solutions are presented in Section III. The proposed mobile crowd sensing platform is explained in detail in Section IV. The cost models and the advantages of using the proposed platform are discussed in Section V. Section VI discusses the implementation details. Experimentation and evaluation details are presented in Section VII. Related works are discussed in Section VIII. Finally, Section IX concludes this paper.

II. Problem Definition and Motivation

In Section I, we briefly introduced our problem domain. In this section, we explain the problem we address in this paper in detail.

The mobile crowd sensing technologies are widely used to collect data in different contexts in the IoT paradigm. Due to popularity of Big Data technologies, processing and storing large volumes of data have become easier than ever. However, still such large-scale data management tasks are economically costly. For example, Microsoft Azure[2] cloud computing platform charges 541 USD/month for 8 cores and 14 GB RAM. Google[3] cloud service pricing is similar. Not everyone is interested in such large volumes of data collection and analysis. Further, not everyone has the financial and computational resources to deal with large volumes of data. Therefore, there is a real need for a mobile crowd sensing platform that is capable of capturing sensor data on-demand based on user requests and the conditions imposed by the data consumers.

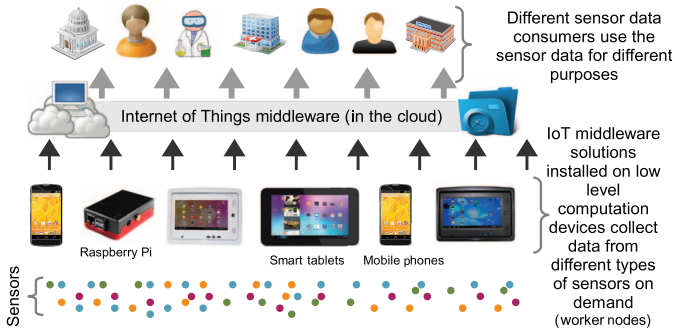


Fig. 1. Proposed platform can be installed on both mobile and static resource-constrained devices. The platform provides easy ways to connect sensors. Each of this platform instances acts as worker nodes and able to carry out sensing tasks as directed by the cloud-based IoT middleware.

Sensing as a service model, as illustrated in Fig. 1, shows how cloud IoT middleware (e.g., GSN [10]) works hand-in-hand with multiple worker nodes (e.g., C-MOSDEN). We identify two fundamental components in this sensing as a service architecture: 1) the cloud platform which manages and supervises the overall sensing tasks and 2) worker nodes that actually perform the sensing tasks as instructed by the cloud IoT platform. It is important to note that our objective is not to analyze the data and extract any knowledge. In this context, our objective here is to collect only the most important and relevant data, so the interested data consumers can use the data to extract the knowledge in an efficient manner with minimum use of computational resources, energy, time, and labor. Our proposed platform is ideal to be installed on worker nodes. Further, IoT middleware platforms such as GSN [10] can be used as the cloud middleware. The cloud IoT middleware evaluates the availabilities of worker nodes and sends the requests to a specific number of selected worker nodes. More importantly, sensor data consumers may impose specific conditions on the data acquisition or transfer, such as *sense only when a certain activity occurs.* The detailed functional requirements of this system are discussed in Section III.

III. Functional Requirement Analysis

In this section, we discuss some of the major functional requirements of a worker node in an ideal on-demand mobile crowd sensing platform. Let us consider three different

[1]It is also important to note that C-MOSDEN is closely integrated into the global sensor network (GSN) cloud middleware [10].

[2][Online]. Available: http://www.windowsazure.com/en-us/pricing/details/virtual-machines/

[3][Online]. Available: https://cloud.google.com/products/app-engine/

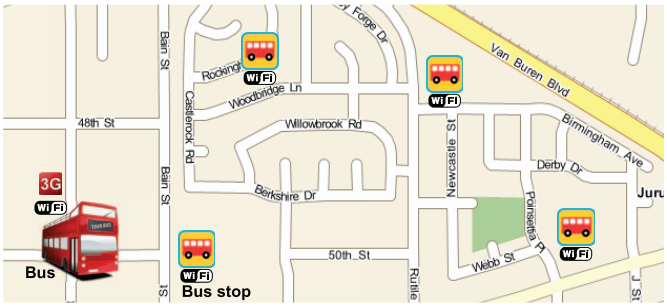


Fig. 2. *Usecase Scenario 1*: Sensors are deployed in buses in a Smart City environment and the data are expected to be collected based on context information and conditions provided by the data consumer.

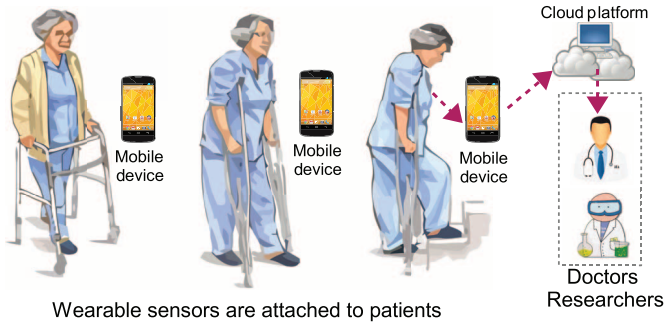


Fig. 3. *Usecase Scenario 2*: Wearable sensors are attached to patient's body. Doctors and researchers are expected to collect data from the sensors based on context information.

scenarios from three different domains: 1) environmental monitoring; 2) physical rehabilitation; and 3) health and well-being.

A. Scenario 1 (Environmental Monitoring)

John, a researcher at the Department of the Environment, is interested in measuring and monitoring the air pollution in cities. John's team has deployed sensor kits in buses. Each of these sensor kits consists of multiple sensors and a communication device with both WiFi and 3G capabilities. John's team has developed an application that processes data collected by these sensor kits. This application consists of a number of different algorithms that analyze and visualize air pollution in the city. However, according to the way that the algorithms are written, John only needs to collect data when the buses are moving. Sensor data captured while the bus is stopped at a bus stop, or in traffic does not add any value. Therefore, John would like to collect sensor data only when the bus is moving. Further, John does not need real-time data in most of the occasions. Therefore, it is sufficient to push the sensor data to the cloud when the bus researches a bus stop. The communication devices fitted in the bus will connect to the bus stop's WiFi and push the data collected since the last bus stop, as illustrated in Fig. 2. However, John is also interested to receive sensor data in real-time when raining. Therefore, when raining, the communication devices need to use 3G to upload the sensor data to the cloud. However, they still need to adhere to the first rule that says *sense only when moving*.

B. Scenario 2 (Rehabilitation)

Robert is a researcher who oversees a number of a rehabilitation facilities around the country where patients with physical disabilities are treated and rehabilitated. Robert is interested in collecting sensor data from sensors worn by patients while they engage in certain activities. Robert has developed an application that requires data collected from wearable sensors [11] only when patients are walking and climbing stairs (as part of the exercise programs recommended by doctors). Wearable sensor kits push data to the patient's smartphone. Each smartphone pushes data to the cloud when it get access to the Internet as illustrated in Fig. 3. It is important to note that Robert is only interested in data collection when the patients are performing certain activities. The blind collection of data during other times may impact negatively in the analysis done by the application Robert has developed. Such data also waste time and resources in the event that Robert has to filter the data he wants from large volume of irrelevant data.

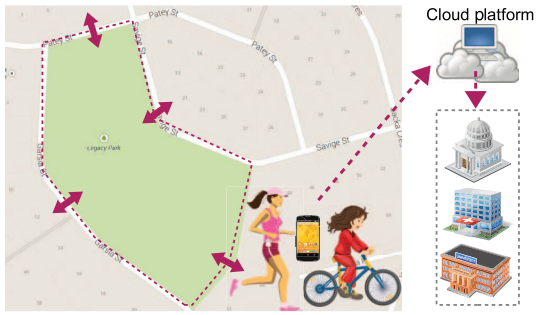


Fig. 4. *Usecase Scenario 3*: Wearable sensors can be used to monitor movements of the general public who use public spaces such as parks for exercising and recreational activities so the authorities can plan further development and upgrades of the infrastructure.

C. Scenario 3 (Health and Well-Being)

Michael is working for the Department of Public Health and Wellbeing. He has been asked to develop a plan to improve the public health in cities by improving the infrastructure that supports exercise and recreational activities (e.g., parks and the paths that support jogging, cycling, and venues for bar exercise). Michael developed a wearable light-weight sensor kit that can monitor a variety of different parameters such as air quality, sound, and movement. Further, Michael has recruited volunteers who are willing to wear those sensor kit when exercising. The sensor kit collects data and pushes it to the volunteer's smartphone. A smart phone application push data to the cloud once it gets connected to the Internet. However, Michael only needs to collect data when a volunteer enters the park areas as illustrated in Fig. 4. Further, Michael only needs to perform sensing only when the volunteers are moving (e.g., walking, running, and cycling). Michael has notices that there is a large amount of people coming to the park during the weekend. In order to reduce the burden to the volunteers, Michael only needs to collect data from a maximum of 30 sensors kits (i.e., volunteers) despite the actual number of volunteers visiting the park in weekends.

As you may have already noticed, each of these scenarios required sensing to be performed under different conditions. Due to limited resources typically, researchers try to limit the data they collect such that the processing can be done with limited resources. Further, due to the ways that analysis tools are programmed, the data they accept may differ. They tend to perform well and produce accurate results when filtered relevant data are provided. The most widely needed filtering conditions in mobile crowd sensing platforms are location awareness (i.e., spatial), activity awareness, time awareness (i.e., temporal), and energy awareness [12]. Therefore, we developed our proposed platform C-MOSDEN to facilitate all of these conditions. Further, it is evident that we require cyber physical systems [13], which have both physical sensing components and cloud-based data analysis software modules, to accommodate the scenarios we described earlier in this section.

IV. Proposed Sensing Platform

In order to address the challenges discussed in Section III, we propose a novel mobile sensing platform called C-MOSDEN. It consists of three components.

1) Context-aware data streaming engine called context-aware mobile sensor data engine (C-MOSDEN). C-MOSDEN platform is based on our previous platform MOSDEN [14]. MOSDEN is an IoT middleware for resource-constrained devices, which allows to collect and process sensor data without programming efforts. Sensing as a service model [7] is a first-class citizen in MOSDEN design. MOSDEN is a client-side tool that can be installed in any android device such as smart phones and DragonBoard 410c. However, MOSDEN does not facilitate any context-aware functionalities. Instead, MOSDEN is a mechanism to filter data streams based on user-defined queries. In C-MOSDEN, new querying capabilities are introduced to support context-aware functionalities.
2) The activity-aware module.
3) A location-aware module.

The complete architecture of the proposed platform C-MOSDEN is presented in Fig. 5. First, we discuss the three main components in the subsequent sections in brief. Then, we briefly introduce the IoT cloud middleware employed, called GSN middleware [10], which is the cloud-based companion platform of the proposed mobile sensing system. At the end, we explain how GSN and C-MOSDEN work together as a system to achieve a common objective.

A. Context-Aware Mobile Sensor Data Engine

C-MOSDEN is a plug-in-based IoT middleware for mobile devices (e.g., mobile phones, tablets, and Raspberry Pi-like platforms) that allows to collect and process sensor data without programming efforts. C-MOSDEN is also a true zero-programming middleware where users do not need to write program code or any other specifications using declarative languages. C-MOSDEN also supports both push and pull data streaming mechanisms as well as centralized and decentralized (e.g., peer-to-peer) data communication.

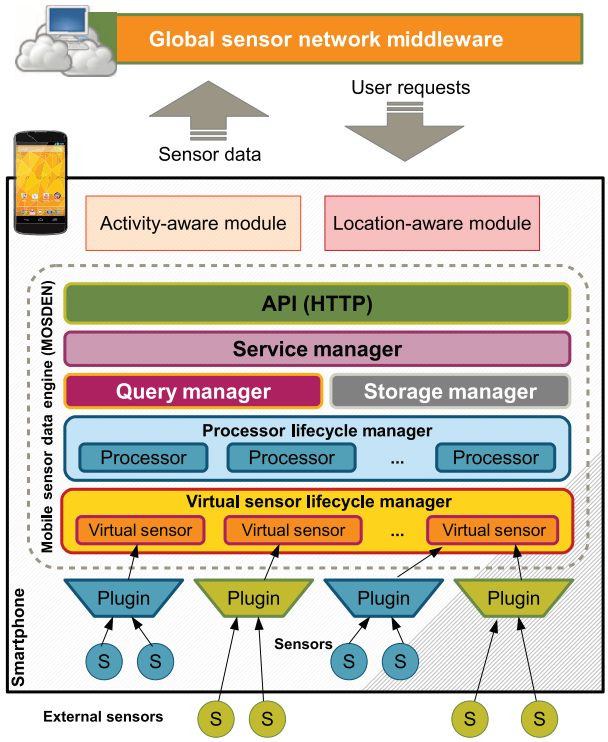


Fig. 5. Proposed C-MOSDEN platform.

Plugins can be installed separately to extend the capabilities of C-MOSDEN (e.g., provide support for difference types of sensors). This engine supports a number data processing and filtering capabilities such as comparison operators, average, etc. Additionally, it supports scalable and distributed sensing. C-MOSDEN can handle more than 100 user requests at a given time. Performance evaluation details are presented in [14] and [15]. In C-MOSDEN, mobile devices are configured using a human readable SQL-like query language as explained in following sections.

B. Activity-Aware Module

The activity-aware module is capable of recognizing six different activities. The detectable activities are: 1) moving in a vehicle; 2) cycling; 3) walking; 4) running; 5) still (not moving); and 6) tilting (falling). Therefore, users can combine these activities to build different types of queries.

C. Location-Aware Module

The location-aware module is capable of recognizing when the device moves into a certain area and moves away from a certain area. These locations are defined using latitude, longitude, and radius in meters. Location awareness can also be combined with other sensing parameters as presented in Fig. 5.

Let us consider three different queries built to support the three scenarios presented in Section III. Fig. 6 illustrates how to combine different sensing parameters including both activity and location awareness.

Mostly, the sample queries are self-descriptive. However, it is important to note that activities are represented by numbers as explained in Section IV-B. Further, "within" location is defined

Scenario 1:

```
SELECT Temperature, Humidity, CO2, CO, NO, SO, Activity FROM
PollusionSensorPlugin, ActivityPlugin WHERE
ActivityPlugin.Activity = 1
```

Scenario 2:

```
SELECT HeartRate, BodyTemperature, ECG, Activity FROM
HealthKitPlugin, ActivityPlugin WHERE ActivityPlugin.Activity = 3
```

Scenario 3:

```
SELECT Temperature, Humidity, CO2, CO, No, SO, Activity FROM
PollusionSensorPlugin, ActivityPlugin, LocationPlugin WHERE
(ActivityPlugin.Activity = 3 OR ActivityPlugin.Activity = 4) AND
LocationPlugin.Within = 1
```

Fig. 6. Sample queries related to the three scenarios presented earlier in this paper. Queries are slightly modified for demonstration and clarity purposes.

as a boolean value (true = 1 and false = 0). Sensor data retrieved by different plugins [14] can be referred in the query. For example, in query 2, health data are retrieved though a plugin called *HealthKit*. This means C-MOSDEN is retrieving data from a external IoT health product [16].

D. GSN Middleware

The GSN [10] is an IoT cloud platform aimed at providing flexible middleware to address the challenges of sensor data integration and distributed query processing. It is a generic data stream processing engine. GSN has gone beyond the traditional sensor network research efforts such as routing, data aggregation, and energy optimization. The design of GSN is based on four basic principles: 1) simplicity; 2) adaptivity; 3) scalability; and 4) light-weight implementation. GSN middleware simplifies the procedure of connecting heterogeneous sensor devices to applications. Specifically, GSN provides the capability to integrate, discover, combine, query, and filter sensor data through a declarative XML-based language and enables zero-programming deployment and management. The GSN is based on a container-based architecture. A detailed explanation is provided in [10]. The *Virtual Sensor* is the key element in the GSN. A virtual sensor can be any kind of data producer, e.g., a real sensor, a wireless camera, a desktop computer, a mobile phone, or any combination of virtual sensors. Typical, a virtual sensor can have multiple input data streams but have only one output data stream. In this work, GSN plays the role of cloud in the IoT platform. It provides the functionality of global scheduler that manages worker nodes. In Section IV-E, we explain how GSN and C-MOSDEN work together as a system.

E. System Work Flow

As illustrated in Fig. 7, first sensor data consumers (e.g., city council, researcher, and medical doctor) submit their requirement. Then, the IoT cloud platform analyses the consumer's problem and decides which sensors are to be used to collect relevant data [17]. Then, the global task scheduler develops a strategic plan on how to delegate the tasks to multiple worker nodes (e.g., C-MOSDEN). Finally, global scheduler sends individual requests to a selected number of worker nodes (e.g., C-MOSDEN). These requests provide exact specifications and sensing objectives on how, when, and where to collect data. Context awareness allows to eliminate significant

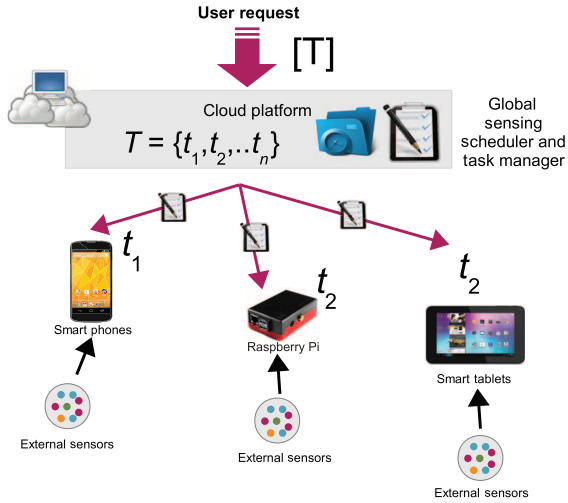


Fig. 7. IoT cloud platform is responsible for global scheduling sensing tasks. For example, GSN performs that task. The IoT cloud platforms can send the specific sensing objectives to the clients, so the C-MOSDEN exactly sends back only the data that are requested.

amount of uninterested data from communicating over limited network resources. Section V will present theoretical models for the resources saving of context-aware selective sensing. Experimental evaluation to justify the theoretical models is discussed in Section VII.

V. Cost Model for Efficient Sensing

In this section, we develop cost models to evaluate the efficiency of C-MOSDEN from three different perspectives: 1) energy; 2) storage; and 3) network communication.

A. Energy Consumption Modeling

The notations will be described as we introduce them in the upcoming sections. As denoted in (1), the total energy consumption of a mobile sensing platform at a given point in time (i.e., Δ) depends on two factors: 1) energy used for computational tasks (denoted by E^{Δ}_{CPU}) and 2) energy used for data communication tasks (denoted by E^{Δ}_{DCom}). It is also important to note that the data communication can also be divided into two parts: 1) ($E^{\Delta}_{\text{DCom}^{\text{S2D}}}$) data communication between sensors (S) and the local computational device (D) (e.g., between external sensors and the mobile phone [18]) and 2) ($E^{\Delta}_{\text{DCom}^{\text{D2C}}}$) data communication between the computational device (D) and the IoT cloud middleware (C) as defined in (2). Further, we can define E^{Δ}_{DCom} based on the communication protocols as well, as presented in (3) (e.g., 3G, WiFi, Bluetooth, ZigBee, and Z-Wave, etc.). Typically, long range protocols consume significantly more energy than short range protocols. However, there are some other energy costs as well (e.g., operating system related computational tasks, display, and so on) where we denote them using a constant K^{Δ} in the following equation:

$$E^{\Delta}_{\text{Total}} = E^{\Delta}_{\text{CPU}} + E^{\Delta}_{\text{DCom}} + K^{\Delta} \tag{1}$$

$$E^{\Delta}_{\text{DCom}} = E^{\Delta}_{\text{DCom}^{\text{S2D}}} + E^{\Delta}_{\text{DCom}^{\text{D2C}}} \tag{2}$$

$$E^{\Delta}_{\text{DCom}} = E^{\Delta}_{\text{3G}} + E^{\Delta}_{\text{WiFi}} + E^{\Delta}_{\text{BT}} + E^{\Delta}_{\text{ZigBee}} + E^{\Delta}_{\text{Z-Wave}}. \tag{3}$$

In Section III, we presented three different scenarios. Each of these scenarios had their own set of requirements and sensing objectives regarding how, what, and when to collect data. A scenario can be considered as an experiment that takes place within a certain period of time. We use S^{Θ} to denote a scenario. Equation (4) defines the total energy consumption by the scenario S^{Θ}

$$E^{S^{\Theta}}_{\text{Total}} = E^{S^{\Theta}}_{\text{CPU}} + E^{S^{\Theta}}_{\text{DCom}} + K^{S^{\Theta}}. \tag{4}$$

In (5), we introduce S^{Ψ} instead of S^{Θ}. In the scenario defined in (4), sensor data are collected using noncontext-aware fashion. This means that the mobile sensing platform has been configured to collect data during the total time of the experiment (no activity-aware or location-aware capabilities have been used). In contrast, (5) defines the total energy consumption when context-aware capabilities are activated. As mentioned earlier, the context-aware capabilities are provided by the C-MOSDEN platforms at some cost. For example, in order to provide activity-aware and location-aware services, C-MOSDEN needs to perform some additional computations. Such additional computations need to be added to the total energy consumption equation. We use $E^{S^{\Psi}}_{\Omega}$ to denote such overhead computational costs

$$E^{S^{\Psi}}_{\text{Total}} = E^{S^{\Psi}}_{\text{CPU}} + E^{S^{\Psi}}_{\text{DCom}} + K^{S^{\Psi}} + E^{S^{\Psi}}_{\Omega}. \tag{5}$$

The total energy consumption by the CPU when context-aware capabilities are not in use (i.e., scenario S^{Θ}) denoted by (6). $\text{PT}^{S^{\Theta}}_{\text{CPU}}$ denotes the CPU processing time of the scenario S^{Θ}. E^{Δ}_{CPU} denotes the energy cost at a given time. Therefore, total energy consumption by the CPU during a scenario S^{Θ} is denoted by $E^{S^{\Theta}}_{\text{CPU}}$

$$E^{S^{\Theta}}_{\text{CPU}} = E^{\Delta}_{\text{CPU}} \times \text{PT}^{S^{\Theta}}_{\text{CPU}}. \tag{6}$$

Similarly, (7) denotes the total energy cost for data communication. At a given point of time, data communication energy cost is E^{Δ}_{DCom}. Total data transmission time is denoted by $\text{TT}^{S^{\Theta}}_{\text{DCom}}$

$$E^{S^{\Theta}}_{\text{DCom}} = E^{\Delta}_{\text{DCom}} \times \text{TT}^{S^{\Theta}}_{\text{DCom}}. \tag{7}$$

It is important to note that in scenario S^{Θ}, the data communication is performed throughout the total duration (due to nonselective, noncontext-aware sensing strategy).

For example, let us consider data communication related energy consumption. $\text{TT}^{S^{\Theta}}_{\text{DCom}}$ is denoted by (8). The total duration of the scenario is denoted by $T^{S^{\Theta}}_{\text{TD}}$. The network communication frequency (i.e., how frequently the data need to be sent to the cloud) is denoted by $T^{S^{\Theta}}_{\text{NCF}}$. Therefore, the number of time that the mobile device needs to push data to the cloud is denoted by $\frac{T^{S^{\Theta}}_{\text{TD}}}{T^{S^{\Theta}}_{\text{NCF}}}$. T^{α}_{DCom} denotes the time it takes to push data to the cloud for one single round. It is important to note that we mainly consider the data communication between the local computational device and the IoT cloud (i.e., $E^{\Delta}_{\text{DCom}^{\text{D2C}}}$) due to its significance over $E^{\Delta}_{\text{DCom}^{\text{S2D}}}$

$$\text{TT}^{S^{\Theta}}_{\text{DCom}} = \frac{T^{S^{\Theta}}_{\text{TD}}}{T^{S^{\Theta}}_{\text{NCF}}} \times T^{\alpha}_{\text{DCom}}. \tag{8}$$

However, in selective context-aware sensing, data are collected only when required. This means that the mobile sensing platform does not push data to the cloud all the time (i.e., T_{TD}). As shown in (9), actual running time (i.e., T_{ART}) is less than the total duration of the scenario (i.e., T_{TD}) due to $T_{\mathbf{\Pi}}$. $T_{\mathbf{\Pi}}$ denotes the time period where mobile sensing platform is not interested to push data to the cloud based on the sensing objectives and instructions provided to it (e.g., through context-aware policies)

$$\downarrow T_{\text{ART}} = T_{\text{TD}} - T_{\Pi} \uparrow. \tag{9}$$

As a result, actual running time (T_{ART}) is less than total duration of a given scenario (T_{TD}) as shown in the following equation:

$$T_{\text{ART}} < T_{\text{TD}}. \tag{10}$$

Then, we can define the energy consumption related to data communication ($E^{S^{\Psi}}_{\text{DCom}}$) for a given scenario Ψ which employs context-aware capabilities to reduce energy wastage as

$$\text{TT}^{S^{\Psi}}_{\text{DCom}} = \frac{T^{S^{\Psi}}_{\text{ART}}}{T^{S^{\Psi}}_{\text{NCF}}} \times T^{\alpha}_{\text{DCom}} \tag{11}$$

$$E^{S^{\Psi}}_{\text{DCom}} = E^{\Delta}_{\text{DCom}} \times \text{TT}^{S^{\Psi}}_{\text{DCom}}. \tag{12}$$

Finally, by applying (12)–(5) and (4), we can model the total saving as defined in the following equation:

$$\text{Total energy cost saving} = \frac{E^{S^{\Psi}}_{\text{Total}}}{E^{S^{\Theta}}_{\text{Total}}}. \tag{13}$$

B. Storage Consumption Modeling

In a similar way to energy consumption modeling, we can also model storage consumption. In a noncontext-aware Θ scenario, the total number of sensor data records collected are denoted by $N^{\Theta}_{\text{Records}}$. Storage frequency is denoted by $T^{S^{\Theta}}_{\text{SF}}$ and the total duration of the scenario Θ is denoted by $T^{S^{\Theta}}_{\text{TD}}$. Therefore, (14) defines the number of records that will be stored during the scenario Θ

$$N^{\Theta}_{\text{Records}} = \frac{T^{S^{\Theta}}_{\text{TD}}}{T^{S^{\Theta}}_{\text{SF}}}. \tag{14}$$

The total storage requirement is defined in (15). It can be calculated by multiplying the number of records need to be stored and the storage requirement of a single record (i.e., $S^{\alpha}_{\text{Record}}$)

$$S^{S^{\Theta}}_{\text{Total}} = N^{\Theta}_{\text{Records}} \times S^{\alpha}_{\text{Record}}. \tag{15}$$

In (9), we showed the reduction of actual running time of a context-aware scenario (i.e., Ψ). This means that the mobile sensing platform now needs to store less number of records compared to a Θ scenario as defined in the following equation:

$$N^{\Psi}_{\text{Records}} = \frac{T^{S^{\Psi}}_{\text{ART}}}{T^{S^{\Psi}}_{\text{SF}}}. \tag{16}$$

As a result, less number of records require less amount of storage as denoted in (17). Finally, we can model the storage cost savings as in (18)

$$S^{S^{\Psi}}_{\text{Total}} = N^{\Psi}_{\text{Records}} \times S^{\alpha}_{\text{Record}} \tag{17}$$

$$\text{Total storage cost saving} = \frac{S_{\text{Total}}^{S^{\Psi}}}{S_{\text{Total}}^{S^{\Theta}}}. \tag{18}$$

C. Network Communication Modeling

In Section V-B, we showed how context-aware Ψ scenarios can reduce storage consumption. Based on that, we base our argument that network communication mostly has the same characteristics as storage. This means that more data we save, it costs more to transfer them to the IoT cloud. Based on (19)–(22), we can infer that context-aware capabilities lead to reduce network communication wastage

$$\text{NC}_{\text{Total}}^{S^{\Theta}} \approx S_{\text{Total}}^{S^{\Theta}} \tag{19}$$

$$\text{NC}_{\text{Total}}^{S^{\Psi}} \approx S_{\text{Total}}^{S^{\Psi}} \tag{20}$$

$$S_{\text{Total}}^{S^{\Psi}} < S_{\text{Total}}^{S^{\Theta}} \tag{21}$$

$$\text{NC}_{\text{Total}}^{S^{\Psi}} < \text{NC}_{\text{Total}}^{S^{\Theta}}. \tag{22}$$

Finally, (23) models the total network communication savings of C-MOSDEN when employing context-aware capabilities.

$$\text{Total network communication saving} = \frac{\text{NC}_{\text{Total}}^{S^{\Psi}}}{\text{NC}_{\text{Total}}^{S^{\Theta}}}. \tag{23}$$

In the above three sections, we theoretically explained how and why context-aware selective sensing is more efficient over nonselective noncontext-aware sensing strategies. In Sections VI and VII, we validate the theoretical modeling by conducting a series of experimental evaluations.

VI. Implementation and Experimental Test-Bed

This proposed context-aware mobile sensing platform C-MOSDEN is developed using the Android platform. We used a Google Nexus 4 device with the Android KitKat operating system for evaluations. Hardware of the device consists of Qualcomm Snapdragon S4 Pro CPU, 2 GB RAM, and 16 GB storage. In order to verify the hypothesis and the cost models, we ran different usecase experiments as explained in Section VII. Each usecase is run ten times and average has been presented as the results. We employed a human actor to perform the three scenarios mentioned in Section III. More importantly, we also ran each usecase under different configurations (e.g., different numbers of sensors used depending on the experiment) to create a benchmark for comparison. We also ran the experiment with and without context-aware capabilities. We built the current activity-aware module using the Android SDK application programming interface (API). We also created a location-aware module by using geofencing technique provided in Google API.[4] It is important to highlight that our intention is not to develop a more efficient activity recognition or geofencing technique, but to use activity and location awareness to enable selective sensing in sensing as a service domain toward improving efficiency. Furthermore, it is very easy to replace the context-aware module that will be employing at a given time. Thus, regardless of the underlying context-aware modules used, the system will always behave the same way from the user perspective. We used a location mocking tool called *fake GPS location* to test geofencing capabilities. In all the evaluations, CPU usage (consumption) is measured in units of jiffies.[5]

[4][Online]. Available: https://developer.android.com/reference/com/google/android/gms/location/package-summary.html

VII. Lessons Learned and Discussion

In this section, we present details of the experimental evaluations that are performed using the proposed sensing platform C-MOSDEN. Our experiments consists of both real-world experiments and simulated lab-based experiments.

In the first three series of experiments, our intention was to understand the impact of context-aware functionality toward CPU consumption, memory consumption, and energy consumption. As we highlighted in the cost models, presented in Section V, there are overheads created by activity-aware and location-aware capabilities.

First, we measured the above-mentioned parameters while context-aware capabilities are deactivated and one the accelerometer sensor is configured to collect data. Second, we activated all the sensors available in the smart phone (i.e., accelerometer, gravity, gyroscope, liner acceleration, rotation vector, light, pressure, magnetic fields, orientation, and proximity). We also kept the context-aware capabilities deactivated. Third, we activated the context-aware capabilities and kept the number of sensors used to collect data the same as before. In this case, we activated the context-aware capabilities, however, did not use the capabilities in action. We configured the mobile device to push all the sensed data to the IoT cloud without any context-aware filtering. By doing so, we were aimed to compare the computational requirements with and without context-aware capabilities activated to identify the overhead created by the context-aware reasoning modules.

In these experiments, our objective is to understand how much more computational resources are required by C-MOSDEN when context-aware capabilities are in action. Specially, we changed the number of sensors used to collect data in two different experiments, in order to compare the resource consumption variability when different numbers of sensors are activated, in comparison to when the context-aware capabilities are activated. It is also important to note that, in these experiments, we configured C-MOSDEN to collect sensor data and push to the IoT cloud middleware in 1 s intervals. We ran the experiments for 30 min. The results presented in Fig. 8(a) (CPU consumption), Fig. 8(b) (memory consumption), and Fig. 8(c) (energy consumption).

According to the results, it is evident that context-aware functionalities create some overhead in terms of CPU, memory, and energy. Based on our experience, in this paper as well as in the past [14], CPU and energy consumptions are not very good indicators to measure the computational complexity, specially in Android, due to its autoload balancing of computational requirement between different applications. However, memory

[5]In computing, a jiffy is the duration of one tick of the system timer interrupt. It is not an absolute time interval unit, since its duration depends on the clock interrupt frequency of the particular hardware platform.

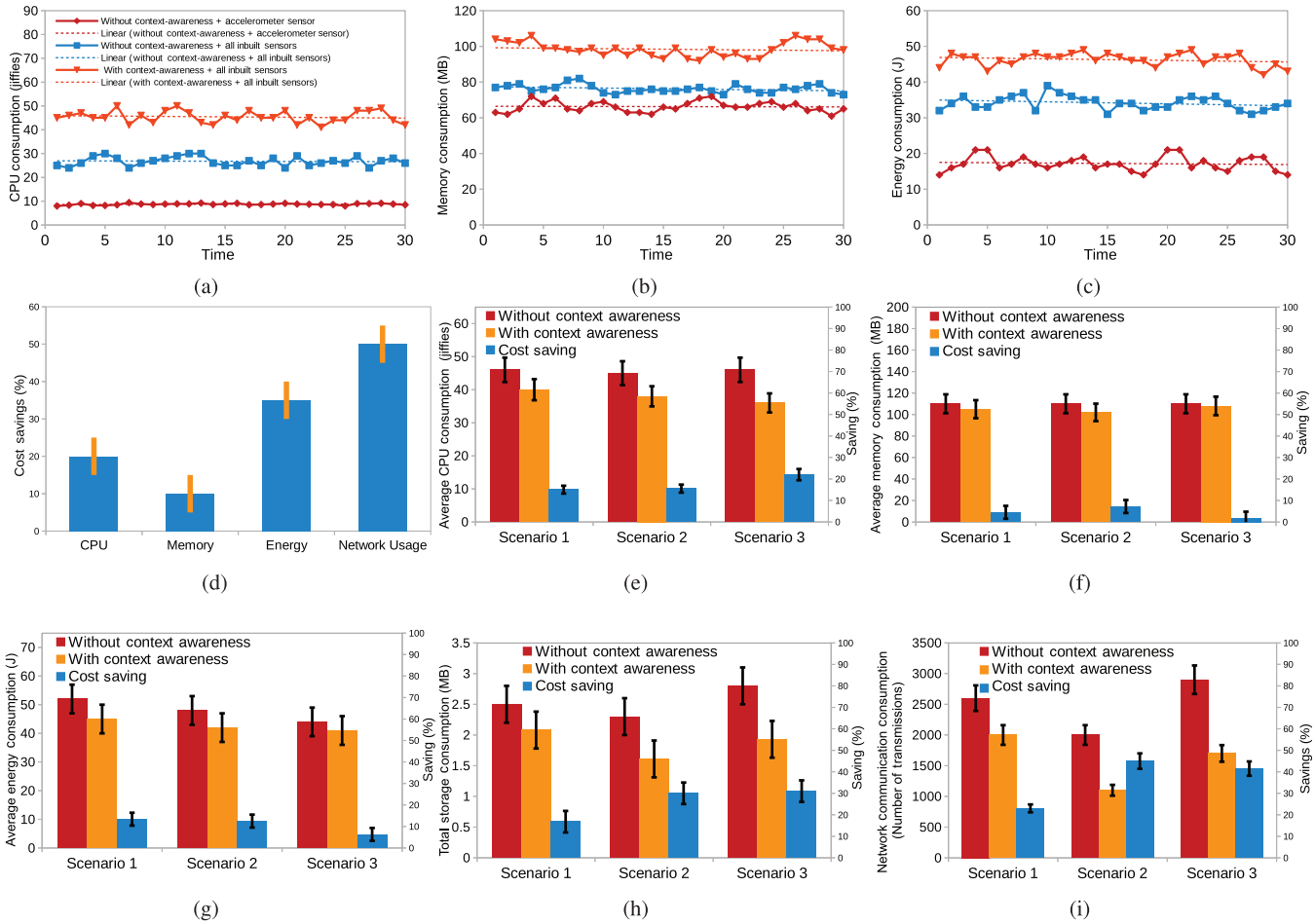


Fig. 8. C-MOSDEN performance evaluation.

is a much better indicator to measure the computational complexity. Android allocates memory less greedily to application as long as it has abundant amount of memory which is 2 GB in Nexus 4 device. If we analyze the memory consumption results in Fig. 8(b), it is evident that, additional overhead created by context-aware functionalities is not substantial.

Next, we ran an experiment to test the capabilities of C-MOSDEN in the real world. In this experiment, we measured CPU, memory, energy consumptions, and network usage. In order to plot all the results in a single graph, we used the cost savings as the common measurements (as a percentage). The results are presented in Fig. 8(d). In these experiments, first, the user walked for 10 min. Then, he cycled for 20 min and then walked again for another 10 min. We ran the experiment with both while context-aware capabilities were ON and OFF. Mobile phone has been configured to collect sensor data using all available sensors. The objective was to collect sensor data only when the user is cycling.

According to the results presented in Fig. 8(d), it is evident that context-aware capabilities have been able to save costs in terms of all four parameters we measured. However, the most significant saving is energy and network usage. Both energy and network usage have been significantly reduced due to selective sensing. Energy consumption is reduced due to the less usage of wireless communication radios [19].

It is important to note that running these experiments required significant amount of time and budget. Therefore, we decided to simulate the scenario we presented earlier in this paper in a lab environment. However, the result we gathered in this real-world example substantially validates our lab simulations. Later, we ran a series of experiments to evaluate three usecase scenarios presented in Section III. We simulated those scenarios in a lab environment. The data collecting specifications are as follows. All the available sensors were configured to collect data in the following experiments. We conducted the experiment both with and without context-aware capabilities activated. We measured CPU, memory, energy, storage, and network consumption. Results are presented in Fig. 8(d)–(i), respectively. To run these experiments, we created predefined data set that simulated the relevant use behavior including location changes and activities changes over time.

1) *Scenario 1 (environmental monitoring)*: Bus moves 5 min and stops for 2 min. This pattern will continue for 60 min (i.e., 10 stops). The sensing objective is to collect sensor data only when bus is moving. The total duration of the experiment is 60 min.
2) *Scenario 2 (rehabilitation)*: The patient performs medically recommended walking exercises for 20 min and rest for 15 min. Then, the patient again walks for 15 min. The sensing objective is to collect sensor data only when the

patient is walking. The total duration of the experiment is 50 min.

3) *Scenario 3 (health and well-being)*: The user cycles to the jogging path for 10 min and then she jogs for 30 min. Next, she does some bar exercise for 15 min before return home by cycling (another 10 min). The sensing objective is to collect sensor data only when user is jogging in the jogging path. The total duration of the experiment is 65 min.

According to the results presented in Fig. 8(e)–(i), it is evident that context-aware capabilities can save costs at different levels depending on the scenario, sensing objectives, conditions, and characteristics. Based on the results gathered in these experiments, we can conclude that any kind of context-aware functionalities (e.g., time awareness and social awareness) that would reduce the uninterested data collection and transmission can be helpful to save costs.

In general, wireless communication radios switching on and off consume significant amount of energy. If the number of times these radios switched on can be reduced, it helps to significantly reduce the energy consumptions. As shown in theoretical models, lesser the amount of data is captured, the less time it will take to transfer the data over the cloud, so the communication radios will only be required for shorter durations. When wireless radios are not actively transmitting data, they will also put less workload on the CPU as well due to less reads/writes from the storage (which also requires less memory). By conducting a number of experiments, we have comprehensively validated the theoretical models presented in Section III. We have also verified the importance of context-aware capabilities integrated into mobile sensing platforms in order to breakdown Big Data into small data, so anyone can analyze them and derive knowledge easily with less amount of resources and budgets.

VIII. Related Work

Mobile phone-based sensing algorithms, approaches, and applications are discussed in [8]. DAM4GSN [20] is an approach based on GSN that is capable of collecting data from internal sensors of a mobile phone and sending it to the GSN middleware. No processing capabilities are provided at the mobile phone end. Therefore, all the information sensed is sent to the server. This approach is inefficient due to the continuous usage of the communication radio of the mobile phone and may also communicate sensor data that are not required or important to the data sensor data consumer [20]. *Dynamix* [21] is a plug-and-play context framework for Android. *Dynamix* automatically discovers, downloads, and installs the plug-ins needed for a given context sensing task. *Dynamix* is a standalone application and it tries to understand new environments by using pluggable context discovery and reasoning mechanisms. Context discovery is the main functionality in *Dynamix*.

One of the most popular type of processing in mobile is activity recognition. Yan *et al.* [22] have presented an energy-efficient continuous activity recognition on mobile phones. Choudhury *et al.* [23] have also developed custom mobile sensing hardware platform for activity recognition. Activities such as walking, running, taking stairs up/down, taking elevator up/down, cooking, working on computer, eating, watching TV, talking, cycling, using an elliptical trainer, and using a stair machine can be detected by using the device. Choudhury *et al.* have used sensors such as microphone, light, three-axis digital accelerometer, barometer temperature, IR, and visible+IR light, humidity/temperature, Compass, 3-D magnetometers, 3-D gyroscope, and 3-D compass to collect data to support their algorithms that detect the activities. Lee *et al.* [24] have developed a similarity system. However, instead of processing the data in the mobile device, it sends data to the cloud by using a smartphone as an intermediate gateway device. Another similar approach has been presented by Laukkarinen *et al.* [25]. They have implemented a distributed middleware for 8-bit microcontroller nodes where executing instructions (e.g., for data processing and event detection) are sent to each node using a *process description language* (PDL). It is important to note that all these approaches focus on building activity recognition modules. In contrast, we employ an activity recognition module to filter unnecessary data processing and communication with the intention of reducing all costs. CONSORTS-S [26] has also used a similar approach. Instead of getting data from external sensors directly into mobile phones, CONSORTS-S uses a custom-made sensor board that connects to the mobile phone using a serial cable which allows the mobile phone to collect data from external sensors.

Most mobile sensing applications can be classified into *personal* and *community sensing* [8]. *Personal sensing* applications focus on the individuals. On the contrary, *community sensing* also termed *opportunistic/crowdsensing*[6] takes advantage of a population of individuals to measure large-scale phenomenon that cannot be measured using single individual. In most cases, the population of individuals participating in *crowdsensing* applications share a common goal. To date, most efforts to develop *crowdsensing* applications have focused on building monolithic mobile applications that are built for specific requirements [27]. Furthermore, the sensed data generated by the application are often available only within the closed population [28]. However, to realize the greater vision of a collaborative mobile *crowdsensing* application, we would need a common platform that facilitates easy development and deployment of collaborative crowd-sensed applications [29].

Grid-M [30] is a platform for lightweight grid computing. It is tailored for embedded and mobile computing devices. The middleware is built using Java 2 Micro Edition, and an API is provided to connect Java-developed applications in a Grid Computing environment. This paper highlights the importance of providing and API-based communication channel, which enables communication. As illustrated in Fig. 1, mobile nodes work similar to grid computing, where they work together to collect sensor data as instructed by the cloud-based IoT middleware or by their own peers (e.g., other mobile sensing platform nodes). Zhang *et al.* [31] have developed a middleware on top of TinyOS (tinyos.net) for TelosB sensors. The data fusion components are designed as agents which they migrate from one node

[6]In this section, we use the terms *opportunistic sensing*, *crowdsensing*, and *participatory sensing* synonymously.

to another. Such migration is an efficient technique in terms of resource utilization. Data fusion consumes the resources only when a given node required to process data. Otherwise, the agents move on to another node on demand. We simulated such behavior in C-MOSDEN where plugins are installed when needed and uninstall when not needed. Another agent-based sensing platform has been proposed by Sun and Nakata [32]. Budde *et al.* [33] have proposed a framework that allows to discover smart objects in the IoT. The framework allows smart objects and services to be registered by providing metadata where it later allows searching and selection. Mori *et al.* [34] have proposed a cloud-based mobile phone sensing middleware [35] that can collectively sense the environment as group of participants. However, if there are more participants present in a given region that expected, the task will be selectively assigned to the most appropriate participants by considering context information such as remaining energy, exact location, and so on. Their approach is also focusing on reducing unnecessary amount of data capturing and communication.

NORS [36] is an open-source platform that enables participatory sensing using mobile phones. It mainly focuses on collecting data instead of processing. The platform includes external sensors, mobile phones, and a cloud service for data storage. Sharing data among mobile phones is not supported. In contract, C-MOSDEN is capable of peer-to-peer communication as well as cloud-based communication. USense [37] is client-side middleware that opportunistically and passively (i.e., without human intervention) performance sensing tasks in crowd sensing fashion. It uses XML definitions to explain a *moment* where the middleware needs to start sensing and stop sensing. The *moment* is composed with a bunch of condition such as location, time, and so on. Similarly, SENSESATION [38] also gathers and stores sensor information using mobile phones and make them directly accessible over the Internet via RESTful web services. Patti *et al.* [39] have proposed an energy-efficient middleware aims at improving energy efficiency of public buildings and spaces exploiting both event-driven and user centric approaches. In their work, sensors are used to detect user presence. Then, system actuates heating systems according to reduce energy wastage.

IX. Conclusion and Future Work

We have presented our C-MOSDEN platform to support on-demand distributed mobile crowd sensing. Our objective was to built a platform that can perform sensing tasks in a collaborative and selective manner. For example, the C-MOSDEN platform can be remotely configured to sense only when a certain activity occurs (e.g., driving, running, and walking). Further, the C-MOSDEN platform supports location-aware sensing (e.g., sense only when a user enters to a particular building). Moreover, the platform has the capability to autonomously select which communication channel (e.g., WiFi or 3G) to use to send the data to the cloud based on context information such as battery level and availability. The proposed platform collects only the data that are relevant to the data consumers, thereby reducing the data storage requirements and processing requirements. We discussed three different real-world usecase scenarios where the proposed platform can offer significant advantages. It was shown to facilitate the efficient and effective mobile crowd sensing functionality at a minimum cost. Through a series of experimentations and evaluations, we showed the importance of selective sensing through the reduction of computational requirements. In general, through selective sensing, we were able to successfully reduce the energy consumption, network communication requirements, and storage requirements. Although the context-aware functionalities have generated a small amount of overhead, it was revealed that the cost savings and benefits far outweighed the increased complexity. In future works, we are planning to enrich C-MOSDEN with privacy-preserving data analytics capabilities.

References

[1] C. Perera, C. Liu, S. Jayawardena, and M. Chen, "A survey on internet of things from industrial market perspective," *IEEE Access*, vol. 2, pp. 1660–1679, 2014.

[2] Y. Zhang, R. Yu, S. Xie, W. Yao, Y. Xiao, and M. Guizani, "Home M2M networks: Architectures, standards, and QoS improvement," *IEEE Commun. Mag.*, vol. 49, no. 4, pp. 44–52, Apr. 2011.

[3] R. Yu, Y. Zhang, L. Song, and W. Yao, "Joint optimization of power, packet forwarding and reliability in MIMO wireless sensor networks," *Mobile Netw. Appl.*, vol. 16, no. 6, pp. 760–770, 2011.

[4] A. Zaslavsky, C. Perera, and D. Georgakopoulos, "Sensing as a service and big data," in *Proc. Int. Conf. Adv. Cloud Comput. (ACC)*, Bangalore, India, Jul. 2012, pp. 21–29.

[5] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelffle, "Vision and challenges for realising the Internet of Things," Eur. Commission Inf. Soc. Media, Tech. Rep. 978-92-79-15088-3, Mar. 2010 [Online]. Available: http://www.internet-of-things-research.eu/pdf/IoT_Clusterbook_March_2010.pdf, accessed on Oct. 10, 2011.

[6] F. Xu, F. Liu, H. Jin, and A. Vasilakos, "Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions," *Proc. IEEE*, vol. 102, no. 1, pp. 11–31, Jan. 2014.

[7] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Sensing as a service model for smart cities supported by Internet of Things," *Trans. Emerging Telecommun. Technol.*, vol. 25, no. 1, pp. 81–93, 2014.

[8] N. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. Campbell, "A survey of mobile phone sensing," *IEEE Commun. Mag.*, vol. 48, no. 9, pp. 140–150, Sep. 2010.

[9] R. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: Current state and future challenges," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 32–39, Nov. 2011.

[10] K. Aberer, M. Hauswirth, and A. Salehi, "Infrastructure for data processing in large-scale interconnected sensor networks," in *Proc. Int. Conf. Mobile Data Manag.*, May 2007, pp. 198–205.

[11] G. Fortino, G. Di Fatta, M. Pathan, and A. Vasilakos, "Cloud-assisted body area networks: State-of-the-art and future challenges," *Wireless Netw.*, vol. 20, no. 7, pp. 1925–1938, April 2014.

[12] A. Huertas Celdran, F. Garcia Clemente, M. Gil Perez, and G. Martinez Perez, "SeCoMan: A semantic-aware policy framework for developing privacy-preserving and context-aware smart applications," *IEEE Syst. J.*, to be published.

[13] S. Khaitan and J. McCalley, "Design techniques and applications of cyberphysical systems: A survey," *IEEE Syst. J.*, to be published.

[14] C. Perera, P. P. Jayaraman, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "MOSDEN: An Internet of Things middleware for resource constrained mobile devices," in *Proc. 47th Hawaii Int. Conf. Syst. Sci. (HICSS)*, Kona, HI, USA, Jan. 2014, pp. 1053–1062.

[15] P. P. Jayaraman, C. Perera, D. Georgakopoulos, and A. Zaslavsky, "Efficient opportunistic sensing using mobile collaborative platform," in *Proc. 9th IEEE Int. Conf. Collab. Comput.: Netw. Appl. Worksharing (COLLABORATECOM)*, Austin, TX, USA, Oct. 2013, pp. 77–86.

[16] C. Perera, C. Liu, and S. Jayawardena, "The emerging Internet of Things marketplace from an industrial perspective: A survey," *IEEE Trans. Emerging Topics Comput.*, vol. 3, no. 4, pp. 585–598, Dec. 2015.

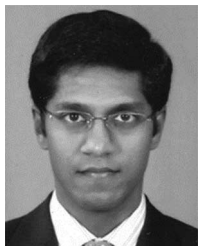
[17] C. Perera, A. Zaslavsky, M. Compton, P. Christen, and D. Georgakopoulos, "Semantic-driven configuration of Internet of Things middleware," in *Proc. 9th Int. Conf. Semant. Knowl. Grids (SKG)*, Beijing, China, Oct. 2013, pp. 66–73.

[18] C. Perera, P. Jayaraman, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context-aware Dynamic Discovery and Configuration of 'Things' in Smart Environments," in *Big Data and Internet of Things: A Roadmap for Smart Environments*. Berlin, Germany: Springer, 2014, vol. 546, ch. 9, pp. 215–241.
[19] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the Internet of Things: Perspectives and challenges," *Wireless Netw.*, vol. 20, no. 8, pp. 2481–2501, Nov. 2014.
[20] C. Perera, A. Zaslavsky, P. Christen, A. Salehi, and D. Georgakopoulos, "Capturing sensor data from mobile phones using global sensor network middleware," in *Proc. IEEE 23rd Int. Symp. Pers. Indoor Mobile Radio Commun. (PIMRC)*, Sydney, Australia, Sep. 2012, pp. 24–29.
[21] D. Carlson and A. Schrader, "Dynamix: An open plug-and-play context framework for android," in *Proc. 3rd Int. Conf. Internet Things (IOT)*, 2012, pp. 151–158.
[22] Z. Yan, V. Subbaraju, D. Chakraborty, A. Misra, and K. Aberer, "Energy-efficient continuous activity recognition on mobile phones: An activity-adaptive approach," in *Proc. 16th Int. Symp. Wearable Comput. (ISWC)*, 2012, pp. 17–24.
[23] T. Choudhury *et al.*, "The mobile sensing platform: An embedded activity recognition system," *IEEE Pervasive Comput.*, vol. 7, no. 2, pp. 32–41, Apr./Jun. 2008.
[24] W. Lee, B. Priyantha, T. Hart, G. DeJean, Y. Xu, and J. Liu, "The CLEO mobile sensing platform," in *Proc. 10th ACM Conf. Embedded Netw. Sens. Syst. (SenSys'12)*, 2012, pp. 371–372.
[25] T. Laukkarinen, J. Suhonen, and M. Hännikäinen, "An embedded cloud design for Internet-of-Things," *Int. J. Distrib. Sens. Netw.*, vol. 2013, p. 13, 2013.
[26] A. Sashima, Y. Inoue, T. Ikeda, T. Yamashita, and K. Kurumatani, "CONSORTS-S: A mobile sensing platform for context-aware services," in *Proc. Int. Conf. Intell. Sens. Sens. Netw. Inf. Process. (ISSNIP)*, 2008, pp. 417–422.
[27] N. Brouwers and K. Langendoen, "Pogo, a middleware for mobile phone sensing," in *Proc. 13th Int. Middleware Conf.*, 2012, pp. 21–40.
[28] R. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: Current state and future challenges," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 32–39, Nov. 2011.
[29] B. Predic, Z. Yan, J. Eberle, D. Stojanovic, and K. Aberer, "ExposureSense: Integrating daily activities with air quality using mobile participatory sensing," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PERCOM)*, 2013, pp. 303–305.
[30] H. Franke, F. Koch, C. Rolim, C. Westphall, and D. Balen, "Grid-m: Middleware to integrate mobile devices, sensors and grid computing," in *Proc. 3rd Int. Conf. Wireless Mobile Commun. (ICWMC'07)*, 2007, pp. 19–19.
[31] L. Zhang, Q. Wang, and X. Shu, "A mobile-agent-based middleware for wireless sensor networks data fusion," in *Proc. IEEE Instrum. Meas. Technol. Conf. (I2MTC'09)*, 2009, pp. 378–383.
[32] Y. Sun and K. Nakata, "An agent-based architecture for participatory sensing platform," in *Proc. 4th Int. Universal Commun. Symp. (IUCS)*, 2010, pp. 392–400.
[33] M. Budde, M. Berning, M. Busse, T. Miyaki, and M. Beigl, "The TECO envboard: A mobile sensor platform for accurate urban sensing and more," in *Proc. 9th Int. Conf. Netw. Sens. Syst. (INSS)*, 2012, pp. 1–2.
[34] S. Mori, Y.-C. Wang, T. Umedu, A. Hiromori, H. Yamaguchi, and T. Higashino, "Design and architecture of cloud-based mobile phone sensing middleware," in *Proc. 2nd Symp. Netw. Cloud Comput. Appl. (NCCA)*, Dec. 2012, pp. 102–109.
[35] M. Rahimi, J. Ren, C. Liu, A. Vasilakos, and N. Venkatasubramanian, "Mobile cloud computing: A survey, state of art and future directions," *Mobile Netw. Appl.*, vol. 19, no. 2, pp. 133–143, 2014.
[36] D. Trossen and D. Pavel, "NORS: An open source platform to facilitate participatory sensing with mobile phones," in *Proc. 4th Annu. Int. Conf. Mobile Ubiquitous Syst. Netw. Serv. (MobiQuitous)*, 2007, pp. 1–8.
[37] V. Agarwal, N. Banerjee, D. Chakraborty, and S. Mittal, "USense—A smartphone middleware for community sensing," in *Proc. IEEE 14th Int. Conf. Mobile Data Manag. (MDM)*, vol. 1, 2013, pp. 56–65.
[38] A. Shirazi, C. Winkler, and A. Schmidt, "SENSE-SATION: An extensible platform for integration of phones into the web," in *Proc. Internet Things (IOT)*, 2010, pp. 1–8.
[39] E. Patti *et al.*, "Event-driven user-centric middleware for energy-efficient buildings and public spaces," *IEEE Syst. J.*, doi: 10.1109/JSYST.2014.2302750

Charith Perera (S'11–M'14) received the B.Sc. degree (Hons.) in computer science from Staffordshire University, Stoke-on-Trent, U.K., in 2009, the M.B.A. degree in business administration from the University of Wales, Cardiff, U.K., in 2012, and the Ph.D. degree in computer science from Australian National University, Canberra, Australia, in 2014.

He was with the Information Engineering Laboratory, ICT Centre, CSIRO, Marsfield, N.S.W., Australia, and involved in OpenIoT Project which is cofunded by the European Commission under Seventh Framework Program. Currently, he is a Postdoctoral Research Fellow with the Open University, Milton Keynes, U.K. His research interests include Internet of Things, smart cities, sensing as a service, privacy, and sensing middleware architecture.

Dr. Perera is a member of ACM.



Dumidu S. Talagala (S'11–M'14) received the B.Sc.Eng. degree (Hons.) in electronic and telecommunication engineering from the University of Moratuwa, Moratuwa, Sri Lanka, in 2007, and the Ph.D. degree in computer science from Australian National University, Canberra, A.C.T., Australia, in 2013.

From 2007 to 2009, he was an Engineer with the Dialog Axiata PLC, Colombo, Sri Lanka. He is currently a Research Fellow with the Centre for Vision, Speech, and Signal Processing, University of Surrey, Surrey, U.K. His research interests include sound source localization, spatial sound-field reproduction, active noise control, array signal processing, and convex optimization.

Chi Harold Liu (M'10–SM'15) received the Ph.D. degree in electronic engineering from Imperial College, London, U.K., in 2010, and the B.Eng. degree in electronic and information engineering from Tsinghua University, Beijing, China, in 2006.

He was with the IBM T. J. Watson Research Center, Yorktown Heights, NY, USA, and the IBM Research China, Beijing, China, as a Staff Researcher and Project Manager from 2010 to 2013, and a Postdoctoral Researcher with Deutsche Telekom Laboratories, Berlin, Germany, in 2010. He is currently a Full Professor and serving as the Assistant Dean with the School of Software, Beijing Institute of Technology, Beijing, China. Since 2014, he also serves as the Director of Data Science Institute, New York, NY, USA; the Director of the IBM Mainframe Excellence Center, Beijing, China; the Director of the IBM Big Data and Analysis Technology Center, New York, NY, USA; and the Director of the National Laboratory of Data Intelligence for China Light Industry, Beijing, China. His research interests include the Internet-of-Things (IoT), big data analytics, mobile computing, and wireless ad hoc, sensor, and mesh networks.

Julio C. Estrella (S'09–M'10) received the B.Sc., M.Sc., and Ph.D. degrees in computer science from the University of Sao Paulo (USP), Sao Paulo, Brazil, in 2002, 2006, and 2010, respectively.

He has experience in Computer Science with emphasis in Computer Systems Architecture, acting on the following themes: service oriented architectures, web services, performance evaluation, distributed systems, computer networks and computer security. He is currently an Assistant Professor with the Institute of Mathematics and Computer Science (ICMC), USP.