Smartphone Sensing Framework

Shahnawaz Alam, Avik Ghose, Arijit Sinharay, Anirban Dutta Choudhury, and Arpan Pal
TCS Innovation Labs, Kolkata
{shahnawaz.alam, avik.ghose, arijit.sinharay, anirban.duttachoudhury, arpan.pal}@tcs.com

ABSTRACT
In this paper, we propose a novel smartphone sensing framework which allows heterogeneous and concurrent sensor access, energy-efficient task partitioning across smartphone and server, and allows integration in third-party applications following some simple rules and constraints. We demonstrate how the framework is able to reduce power-usage on the smartphone as well as maintains optimum network bandwidth usage, while not sacrificing on user experience and sensor accuracy.

Categories and Subject Descriptors
• Software and its engineering~Client-server architectures

Keywords
Mobile Sensing; System Design; Middle-ware; Frameworks; Heterogeneous Sensing; Mobile Performance;

1. INTRODUCTION
Smartphones have become powerful ecosystem with sensing capabilities and internet readiness that is capable of solving many complex problems. They are now used in many areas like healthcare, industries, social media etc. as various sensing and processing device.

This has increased the demand for writing IoT enabled intelligent smartphones applications across diverse domain. As a result, various complex algorithms which use various sensors are being ported to smartphones as an IoT solution. The most widely used domain being healthcare where several apps are available to estimate various parameters like heart rate, blood pressure, activity etc. All these apps use multitude of built-in sensors of mobile. Accelerometer is used to estimate activity like running, brisk walking, jogging, etc. [1]. In [2], researchers present a method to detect falls using accelerometer data. In SmartEvacTrak [3], researchers used magnetometer and Wi-Fi to locate person's zone in a building floor to assist in emergency evacuation. In recent past years, researcher uses camera and IMU sensor information to construct 3D model of indoor/outdoor environment [4]. In HeartSense [5], researchers uses camera sensor and LED flash to extract PPG (Photoplethysmogram) signals to estimate heart rate and blood pressure of a person. These applications are built as standalone versions using smartphone resources only which make them power hungry. Also while running these standalone apps; few sensors, for example camera, does not allow concurrent access by another application due to locking mechanism [7].

In [8], Aharony et al proposed an Open Sensing Framework (in short Funf), which is an extensible sensing and data processing framework for Android. It provides a minimal set of reusable services for extracting, locally setting, and uploading to remote servers a wide range of sensing activities. However, this framework does not include object reuse, lightweight resource binding which consumes more power and degrades the performance of smartphone. Moreover, it also does not talk about network overhead. In [9], Cardone et al presented Mobile sensing framework (MSF), in which all the pre-processing and computation are done in smartphone itself. It provides heterogeneous mobile sensing applications by masking all low-level technical details in reading and processing raw sensor data but it might take long processing time for complex algorithms to run on many low-end phones widely used, also it does not consider battery performance.

In order to overcome such drawbacks we propose a framework to handle heterogeneous sensors and reduce on-board processing. We envisage that such a framework will ease complexity of developing smartphone based sensing applications. The proposed framework also effectively partitions computing tasks between on-board processing and cloud computing to attain trade-off between network load and CPU overhead of the smartphone, which produces the least energy footprint. More specifically, mobile will handle pre-processing unit which will work on raw sensor data and reduce unnecessary and noisy information or artifacts to create less number of data chunk to be streamed over internet thus reducing network load. We use high level abstraction layer for sensors to support heterogeneous sensing, reducing processing time and power consumption by using cloud as a high end processor. We have produced an end-to-end working model of a sensing framework, which we propose to demonstrate in our current work.

2. IMPLEMENTATION DETAILS
Our proposed framework provides users with a platform for developing heterogeneous sensing application by abstracting low level sensor details and processing flow complexity. The framework illustrated in Figure.1 is divided in two units, one is the sensing unit and the other one is the processing and management unit. Sensing unit is present at client side, which is an Android smartphone device, whereas the processing and management unit is implemented partitioned across the client and server side. Framework is capable of accepting task requests from 3rd party apps to aid sensing and processing. Our solution provides flexibility to accept requests from multiple apps and process it simultaneously.

The framework's client unit is an Android background service. The server side is a web service layer on Apache Tomcat. 3rd party application needs to send request for the service to the client application framework for acquiring sensor data, pre-processing and streaming it over internet to the server framework application to process a particular task. Server applies respective algorithms on pre-processed data and calculates the output parameters as defined in task provided by 3rd party app, which it either sends back to the framework application or stores it in cloud for historical analysis. Below we describe in details the whole flow of our framework.

MobiCASE 2016, November 30-December 01, Cambridge, Great Britain
Copyright © 2016 EAI
DOI 10.4108/eai.30-11-2016.2267130

2.1 Handling Third-party Applications

Task is initiated by the 3rd party app only in the form of JSON which is sent to the framework by using Android Intent for Inter-App Communication. JSON provides requirements like parameters to be calculated, sensors required, sensor specific parameters, historical storage, duration of sensor data capturing etc. It can have as many tasks to compute various parameters. The 3rd party app should incorporate this process in their app to use our Framework. The format of JSON has to be strictly adhered by every 3rd party app for successful completion of task. An Android intent is also used to interrupt the previously initiated tasks; in such case calculated result is sent back to the 3rd party app. In case of interruption framework stops acquiring raw sensor data for corresponding task and releasing the resources being used.

2.2 Framework – Client App

2.2.1 Extracting Sensor Data

The client framework app receives the task sent by the 3rd party app and JSON in the bundle is parsed to retrieve task and sensor details. According to the task description the required sensors of smartphone are registered to capture raw data with its timestamp. Framework creates uniformity in different sensors data by abstracting it as a common class object independent of the data structure design of sensors information.

2.2.2 Pre-processing

The pre-processing block processes the raw data from sensor to remove noises and artifacts. It consists of various data processing algorithms like band-pass filter, Fast Fourier Transform, FSM and other signal quality checker.

2.2.3 Transmitting Data

Preprocessed data is now serialized into byte format to transmit over TCP network to server. A socket connection is created to the Linux server for seamless data transfer. Single data stream in bytes containing sensor data and task details are then pushed into this socket channel. Same socket will be used to retrieve back the output from server.

2.3 Framework – Server System

2.3.1 Data Accumulation

Server receives the single data stream from client framework application. The de-serialized data is classified into preprocessed sensor data and task which needs to be executed. Individual tasks initiate respective algorithms available on the server. Depending upon sensor type, logical partitioning is done to route sensors data into different sensor queue pool available for processing. Sequencing of sensor data based on corresponding timestamp managed.

2.3.2 Data Processing

The processing unit on server routes required sensors data queue instance to the algorithm, which helps simultaneous access of multiple sensors data by different algorithms. Algorithm then processes the data queue and outputs the results which are transmitted using previously created socket back to the 3rd party app. This completes a single lifecycle of the task in our framework.

3. DEMO

So far, the proposed methodology has been successfully implemented and tested on BP estimation [5] and AcTrak [1] android apps, running on Nexus 5 and LG G3 as client apps. The server application is implemented in Java Enterprise Edition on Apache Tomcat on a Linux system. Trepn Profiler app [6], a 3rd party profiling app for Android, is used to measure key resources like battery consumption, CPU and network load. A demo will consist of showing these resources comparisons between standalone applications and through our proposed methodology. This will clearly outline the advantage of using our proposed framework. The Demo will only require a smartphone, a Linux machine and a Wi-Fi network.

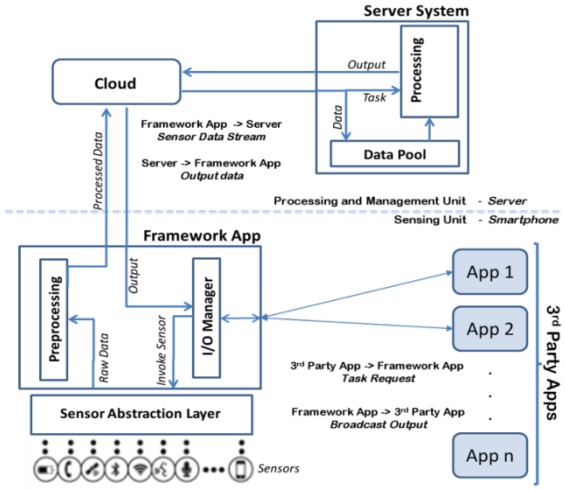

Figure 1. Framework Architecture.

4. REFERENCES

[1] Chandel, Vivek, et al. "AcTrak-unobtrusive activity detection and step counting using smartphones." International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services. Springer International Publishing, 2013.

[2] Chandel, Vivek, et al. "Demo: Exploiting IMU Sensors for IoT Enabled Health Monitoring." Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services Companion. ACM, 2016.

[3] Ahmed, Nasimuddim, et al. "Smartevactrak: A people counting and coarse-level localization solution for efficient evacuation of large buildings." Pervasive Computing and Communication Workshops (PerCom Workshops), 2015 IEEE International Conference on. IEEE, 2015.

[4] A. Malik, et al. "A Multi-Sensor Information Fusion Approach for Efficient 3D Reconstruction in Smart Phone," International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV), 2015.

[5] Banerjee, Rohan, et al. "Noise cleaning and Gaussian modeling of smart phone photoplethysmogram to improve blood pressure estimation." 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2015.

[6] Profiler, Trepn. "Qualcomm."

[7] https://developer.android.com/reference/android/hardware/Camera.html#open%28int%29

[8] N. Aharony and W. Gardner, 2012, Funf Developer Site, http://www.funf.org.

[9] G. Cardone, A. Cirri et al., " MSF: An Efficient Mobile Phone Sensing Framework," International Journal of Distributed Sensor Networks, 2013