An Integrated Cloud-based Framework for Mobile Phone Sensing

Rasool Fakoor
Dept. of Comp. Sci. and Eng.
University of Texas at Arlington
Arlington, Texas, USA
rasool.fakoor@mavs.uta.edu

Mayank Raj
Dept. of Comp. Sci. and Eng.
University of Texas at Arlington
Arlington, Texas, USA
mayank.raj@mavs.uta.edu

Azade Nazi
Dept. of Comp. Sci. and Eng.
University of Texas at Arlington
Arlington, Texas, USA
azade.nazi@mavs.uta.edu

Mario Di Francesco
Dept. of Comp. Sci. and Eng.
Aalto University
Espoo, Finland
mario.di.francesco@aalto.fi

Sajal K. Das
Dept. of Comp. Sci. and Eng.
University of Texas at Arlington
Arlington, Texas, USA
das@uta.edu

ABSTRACT

Nowadays mobile phones are not only communication devices, but also a source of rich sensory data that can be collected and exploited by distributed people-centric sensing applications. Among them, environmental monitoring and emergency response systems can particularly benefit from people-based sensing. Due to the limited resources of mobile devices, sensed data are usually offloaded to the cloud. However, state-of-the art solutions lack a unified approach suitable to support diverse applications, while reducing the energy consumption of the mobile device. In this paper, we specifically address mobile devices as rich sources of multi-modal data collected by users. In this context, we propose an integrated framework for storing, processing and delivering sensed data to people-centric applications deployed in the cloud. Our integrated platform is the foundation of a new delivery model, namely, Mobile Application as a Service (MAaaS), which allows the creation of people-centric applications across different domains, including participatory sensing and mobile social networks. We specifically address a case study represented by an emergency response system for fire detection and alerting. Through a prototype testbed implementation, we show that the proposed framework can reduce the energy consumption of mobile devices, while satisfying the application requirements.

Categories and Subject Descriptors

C.2.4 [**COMPUTER-COMMUNICATION NETWORKS**]: Distributed Systems—*Distributed applications*; C.2.1 [**COMPUTER-COMMUNICATION NETWORKS**]: Network Architecture and Design —*Distributed networks*

MCC'12, August 17, 2012, Helsinki, Finland.
Copyright 2012 ACM 978-1-4503-1519-7/12/08 ...$15.00.

General Terms

Design, Reliability

Keywords

Mobile Cloud Computing, People-centric Applications, Mobile Application as a Service (MAaaS), Publish/Subscribe, Mobile Phone Sensing

1. INTRODUCTION

The proliferation of mobile devices, combined with their diverse sensing capabilities, has made them a ubiquitous and rich source of sensory data. Such data include location (e.g., GPS), orientation (e.g., gyroscope, accelerometer), textual (social) and multimedia (e.g., image, audio and video) data [9]. They can be jointly used to develop a wide range of applications aimed at improving the quality of life. These applications include environmental [12, 3] and traffic monitoring [18, 17], navigation [13] and emergency response systems [10], as well as smart healthcare and well-being [9].

Many of the applications mentioned above are people-centric, in the sense that they assist individual users, a group of people, or even a large community, by leveraging sensory data from a large number of sources [2]. Depending on user demands, sensing mechanisms can be either passive or active, namely, participatory. Even though some of these applications have been deployed in many environments [2, 9], integrating them into a single platform is still an open research problem. This is due to the presence of a large number of applications with contrasting requirements, diverse sensing sources and platforms, as well as multi-modal data.

In this paper, we propose an integrated framework which allows unified sensing for diverse people-centric applications. To ensure scalability, availability and robustness of the sensing and application environments, the framework is defined as part of the cloud infrastructure itself. Specifically, our integrated platform is the foundation of a new delivery model – namely, Mobile Application as a Service (MAaaS) – which allows the creation of people-centric applications across different domains, including participatory sensing and mobile social networks. Each mobile device acts as an independent multi-modal data source, and the proposed cloud-based framework supports and coordinates a large number of users and applications by adapting to the resource requirements of individual applications. Through a case study represented by an emergency response scenario, we show how the framework can be used to enable large-scale people-centric applications. Specifically, we detail a prototype implementation of the framework based on Amazon web services (AWS) [1] and show the energy savings obtained by the proposed framework at the mobile device when multiple applications are deployed at the same time.

The rest of the paper is organized as follows. Section 2 summarizes the related works, while Section 3 introduces the proposed framework. Section 4 presents a case study, which is then used as a reference scenario for the proposed framework. Section 5 details the prototype implementation of the proposed system, while Section 6 provides some experimental results. Finally, Section 7 concludes the paper.

2. BACKGROUND AND MOTIVATION

In this section, we first introduce the reference cloud model, then we present a few examples of people-centric applications. Based on them, we motivate the need for an integrated cloud-based framework for mobile phone sensing.

Mobile Application as a Service (MAaaS) In the MAaaS cloud delivery model, the cloud acts as the host and the medium through which all mobile applications can be accessed. Connectivity to the cloud is provided through a thin mobile client installed on the mobile device. On the other hand, mobile applications hosted on the cloud are accessed through web widgets, i.e., applications which can be installed and used within a web page, thus making them platform independent [14]. Furthermore, the cloud ensures efficient collection, distribution, processing and delivery of the data required by mobile applications. Developers can access the data, develop and deploy their applications by using the Infrastructure as a Service (IaaS) and Platform as a Service (PaaS) cloud delivery models.

People-centric Applications Several cloud-based applications exploited mobile devices in a people-centric fashion. In Biketastic [13], VTrack [17] and the Mobile Millennium project [18], network-based localization and GPS input from mobile devices were exploited to determine the location of users so as to develop a traffic monitoring and navigation system. In the Peir project [12], GPS data from end users were linked to the available information about the visited location in order to provide a report of environmental impact (e.g., pollution) based on the observed mobility pattern. In the Garbage watch [3], Urban Atmospheres [8] and CenceMe [11] projects, users were recruited to actively provide sensory data – such as location, video and audio recordings from the camera and the microphone of the mobile device, as well as textual (updates to social networks) data – for environmental monitoring and for improving people's lifestyle. In [20], GPS and gyroscope data from mobile phones were exploited to discover potholes on roads and alert travelers. In [10], a cloud-based social networking site was developed to facilitate information flow for aid and search as well as damage assessment, enumeration and collaboration in disaster scenarios.

Need for an Integrated Framework As apparent from the previous discussion, several applications share some needs as for the collected sensory data. For instance, location (i.e., GPS) is required by almost all the above-mentioned applications, while gyroscope readings are needed in [13] and [20]. Different applications running on a single mobile device and capturing data individually are not the most efficient choice for the mobile device, in terms of both resource usage and energy consumption. In [15], an architecture for massive urban scanning using vehicular and mobile sensors was presented for reliable data gathering and storage. However, the issue of integrated sensing was not addressed. Essentially, for each application an individual data flow is maintained in the cloud, which increases the stress on the cloud infrastructure when dealing with a large number of users. We overcome this drawback with the help of a unified sensing and data delivery model to the cloud for all the applications involved. Based on the requirements of individual applications, the cloud infrastructure forwards a subset of the data only to the concerned applications. In [7], a framework for connecting wireless sensor networks to the cloud was proposed for several community-based applications. The work focused on developing a publish/subscribe system for efficient distribution of data in the cloud, as well as on promoting cooperation between different cloud providers. In contrast, our focus is to develop a framework for unified sensing, and to provide application developers with a complete solution for robust and flexible deployment of people-centric applications in dynamic environments.

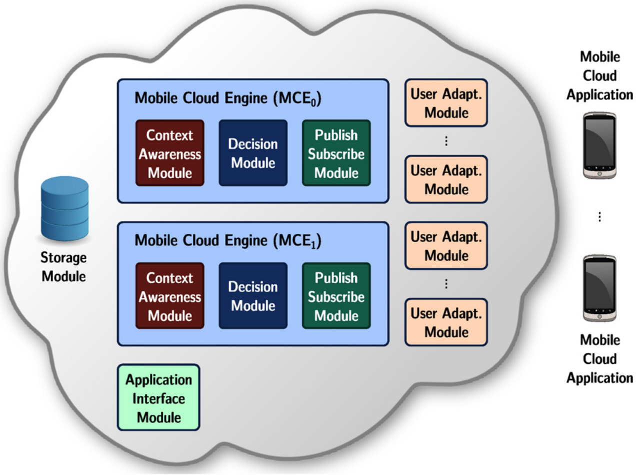

Figure 1: System architecture and components

3. MOBILE SENSING FRAMEWORK

As depicted in Figure 1, the system components of our framework reside at both the mobile device and the cloud infrastructure.

The *Mobile Cloud Application* (MCA) is a thin application that runs on the mobile device. It collects the sensory data from the mobile device by minimizing the resource utilization as well as the energy consumption, and supports both active (i.e., participatory) and passive data collection. The MCA sends the data to the cloud infrastructure, and also interfaces end-users with the cloud applications. The MCA allows the users to register to the system by means of a unique identity, generated by the cloud and associated with the mobile device. End-users can search the applications hosted on the cloud and select those they are interested in directly from the MCA.

The rest of the system components is part of the cloud infrastructure. The *User Adaptation Module* (UAM) interfaces the MCA with the *Mobile Cloud Engine* (MCE). Specifically, there is one UAM for each MCA. When multiple MCEs share a common user (namely, a MCA), the related device sends the data once to the UAM which, in turn, sends a copy to the relevant MCEs.

The *Storage Module* (SM) allows to save and retrieve sensory data. It supports multi-modal data, ranging from scalar data (e.g., location, accelerometer and gyroscope readings) to generic textual information (e.g., summary descriptions uploaded by the users) and multimedia data (i.e., images, audio and videos captured through the mobile device). As a consequence, the SM is data-independent, but provides means to interconnect to the cloud for mining and knowledge extraction purposes. The major features of the SM are scalability and responsiveness.

The *Application Interface Module* (AIM) interacts with the framework to get the required data. It further allows the application developer to access computing and storage resources on-demand for flexibility and scalability.

Finally, the *Mobile Cloud Engine* (MCE) processes and controls the data flow in the framework, and is instantiated for each application. Since the MCE is the core component of our solution, it will be detailed in a separate subsection below.

3.1 Mobile Cloud Engine

Our proposed framework contains two types of mobile cloud engines. The first type is called *control MCE* or (MCE_0). There is only one MCE_0 in the whole framework, independent of the actual people-centric applications. The MCE_0 is responsible to initiate other MCEs when a new application request arrives in the framework. In addition, it always collects the sensory data from mobile users in predefined periods, regardless of the application requests. The data collected by the MCE_0 can be used for different major purposes, namely: mining; processing; and as a baseline to start other MCEs. The second class of MCEs is instantiated based on the actual application requests. These MCEs are mainly employed to deal with specific people-centric applications. However, both classes of MCEs have the same structure to provide coherency in the framework.

The MCE is composed of three sub-modules. The *publish/subscribe* (PS) *module* is primarily responsible for establishing the data flow between UAMs and the MCEs. The PS module is composed of control and data message queues. The control message queue handles all the control messages, such as the data sending rate. The data message queues are composed of multiple queues, each associated with a different priority (e.g., low, medium, and high). The data queue delivers the captured data from the UAM to the associated applications, and also forwards them to the storage module. Furthermore, the PS module delivers the processed information back to the mobile users. In this framework, the PS modules are classified in three categories, according to the related priorities, namely, low, medium, and high. Different priorities are mainly used to regulate the sampling rate at the mobile device, the data reporting rate to the cloud, and the actual data processing within the cloud.

The *decision module* (DM) handles and regulates the transactions among the different parts of the MCE. It also classifies the PS modules based on the requests received from applications. Specifically, once an application request arrives in the MCE, the decision module identifies which type of PS module is needed to satisfy the request. Furthermore, it recognizes the potential group of mobile users who can participate to the application query. Finally, the DM module initiates the context-awareness module (described next) to provide people-centric and location-based services. The decision module provides functions to process and mine data. It can use unsupervised learning approaches to deal with unstructured data in the system. One example is to find a coherent group of mobile phone users who share common interests (e.g., through their frequently visited places) or to predict user behavior.

Finally, the *context-awareness* (CA) *module* provides context information to the different people-centric applications. In its simplest form, the context is represented by location. For example, a pollution monitoring application may consider the pollutant concentration at the locations in the vicinity of the user. In the more general case, context may include the activity of the user or its social interactions with other users.

4. CASE STUDY

Texas is very likely to experience wildfires in Summer. The state of Texas and the federal government spend substantial amount of money to handle wildfires, in order to ensure public safety and reduce damages. For instance, from November 2010 to October 2011, about 30,457 fires spread over 3,993,716 acres and destroyed 3,017 homes [16]. The cost of damages to houses was estimated to be more than 100 million dollars [4]. Dispatching fire fighters to the exact location of wildfires and preventing the fire from spreading play an important role in efficiently handling the emergency. However, placing cameras and sensors at all locations to continuously collect information is not feasible due to the high deployment and maintenance costs.

In this section, we demonstrate how our proposed architecture can help in this scenario. Let us assume that the mobile cloud application for emergency response has already been installed on the user mobile devices and that a fire happens in an certain area. In the proposed framework, user location is periodically updated and stored in the the mobile cloud framework either explicitly by the mobile device user or through a request from the wildfire monitoring and response agency. When a wildfire is reported, the agency initiates a request to the framework with the information of the affected area. After receiving the request, the framework finds the people in that area, based on their GPS locations. Then, the framework sends an emergency notification and requests for updates on the situation.

At the same time, the mobile cloud framework updates the agency with current available data in the system until new data arrives. The application can provide different visualizations, ranging from a map showing the location of people who are in the vicinity to a live video stream of a certain place. By using this method, the agency can obtain an accurate characterization of the wildfire without deploying any camera or sensor. The application can provide indirect communication between the agency and mobile phone users to explicitly obtain additional information which, in turn, can help increase the accuracy and reliability of the decisions made by the agency. The work flow of the proposed framework is described next.

4.1 Work flow

In the following, we will refer to an emergency response system to illustrate the work flow between the different components of the framework. The simplest scenario is represented by the case where there are no applications sending requests to the cloud. Recall from Section 3.1 that the MCE_0 periodically collects data from the registered users regardless of the active applications. As a consequence, sensory data from mobile devices are delivered to the appropriate UAM and then to a low priority queue. The MCE_0 asynchronously polls the queue for new data. Furthermore, the MCE_0 processes the fetched data and forwards them to the storage module.

Let us consider now the case where two agencies corresponding to two different applications – i.e., a fire agency and an emergency healthcare agency – intend to exploit the cloud framework. They initiate two different requests to the system. Note that a new MCE instance is created for each application. For instance, the fire agency needs visual data of the fire or location of users in the vicinity of the location of fire, say X. Then a new MCE_1 instance finds all users in the vicinity of location X and sends them a notification to collect data relevant to the situation. On the other hand, the emergency healthcare agency needs information about physicians and nurses in the vicinity of location Y. To this end, an additional MCE_2 instance finds all the users in the vicinity of location Y then sends them a notification alert.

The step-by-step procedure for handling an individual request is illustrated in Figure 2. First, the agency initiates a request to the corresponding MCE. Second, the MCE sends a request to the storage module to find all registered users in the vicinity of the specified location, e.g., X or Y. After receiving the list of the matching users $U = \{u_1, u_2, .., u_n\}$ from the storage module, the MCE moves them from the low priority queue to a higher priority queue, in order to collect data at a higher rate. The MCS also sets up an individual PS module for sending the data collection notifications to the specific users in U. On the other side, the UAM polls the PS module for notifications and sends the data collected by the users to the relevant applications. Hence, if a user subscribed to different applications, sampled data are just sent once, and the UAM forwards them to the corresponding MCEs. Finally, the MCE polls the queue, fetches the data, and hands them over to the CA and AIM modules, e.g., for visualization purposes. Moreover, it stores these data in the storage module. This procedure continues until the agency requests the termination of the service. Once the service is terminated, users are switched back to the lower priority queues in the MCE_0, and the higher priority queue are released.

5. IMPLEMENTATION

We implemented a prototype system that offers the basic functionalities of the proposed framework. The components residing at the cloud have been realized by using Amazon Web Services (AWS) [1] and Google Fusion Tables [6]. The Mobile Cloud Application (MCA) was built for Android devices running the 2.1 Platform [5] and the AWS SDK for Android [1]. Additional details are provided below.

5.1 Mobile Cloud Engine (MCE)

The *Publish/Subscribe (PS) module* was implemented by using Amazon's Simple Notification System (SNS) and Simple Queue Service (SQS) [1]. SNS allows a cloud application to easily send notifications to the subscribers. SNS exploits the concept of *topic* to broadcast a notification from publishers to subscribers. In this context, the publisher is the application that sends messages, while the subscriber is the application (e.g., the one running on the mobile phone) that can receive the messages. A topic is an access point which defines specific subject or event type for publishing messages and allows clients to subscribe for notifications [1]. In other words, the topic is the connection point between the sender and receivers. The notification can be in the form of SMS, email, SQS, and so on. In our framework, the MCE creates the topic and the mobile client application subscribes (unsubscribes) the mobile phone to (from) the topic. The framework uses the SQS as publisher and subscriber to a topic. SQS is an asynchronous mechanism by which the data or messages can be transferred between distributed components even without losing messages or requiring each component to be always available [1]. For example, the mobile cloud application uploads the GPS data to the SQS and the MCE fetches the data from the SQS when needed at any time. SQS simplifies the interactions between the mobile cloud applications and the MCE. Furthermore, it supports read (and write) operations from (to) a large number of mobile cloud applications. Moreover, it provides security by authentication method. Using SQS in our framework enables mobile phones to send a large number of messages in parallel. In addition, SQS and SNS together enable the MCE and mobile client applications to send messages to a huge number of users in multiple formats quickly and easily. When the MCE needs to send a message to a group of mobile phones, it publishes the message to a topic. Once the messages have been published to a topic, all the subscribers of the SQS get the messages. Since, the mobile client application of each phone regularly polls the SQS, once the message is available in SQS, it can get the message.

The *context-awareness module* is realized with reference to location-based information and is implemented through the Google Fusion Tables API [6]. Google Fusion Tables provide mechanisms for publishing, retrieving, and rendering location-based data. It has been chosen for its features of scalability, accuracy, and availability, as well as for the built-in visualization based on Google Maps. Google Fusion tables are controlled by a Google account such that each account has its own fusion tables. In the proposed framework, the MCE was connected to a Google account in order to be able to use these features. Whenever the MCE has to return the result to the application community in the form of maps, it uses Google Fusion Tables. Specifically, the MCE writes the result into a fusion table, and based on those data, renders the map by using the Google Fusion API. Once the map has been generated, the MCE returns the obtained map and displays it through the widget. In addition, the data in the fusion table is continuously updated by MCE upon arrival of new data. As a result, the module is capable of rendering dynamic information to the map. In the current implementation, the *decision module* is limited to initialize the MCE, instantiate the PS modules, and handle the transactions between other modules.

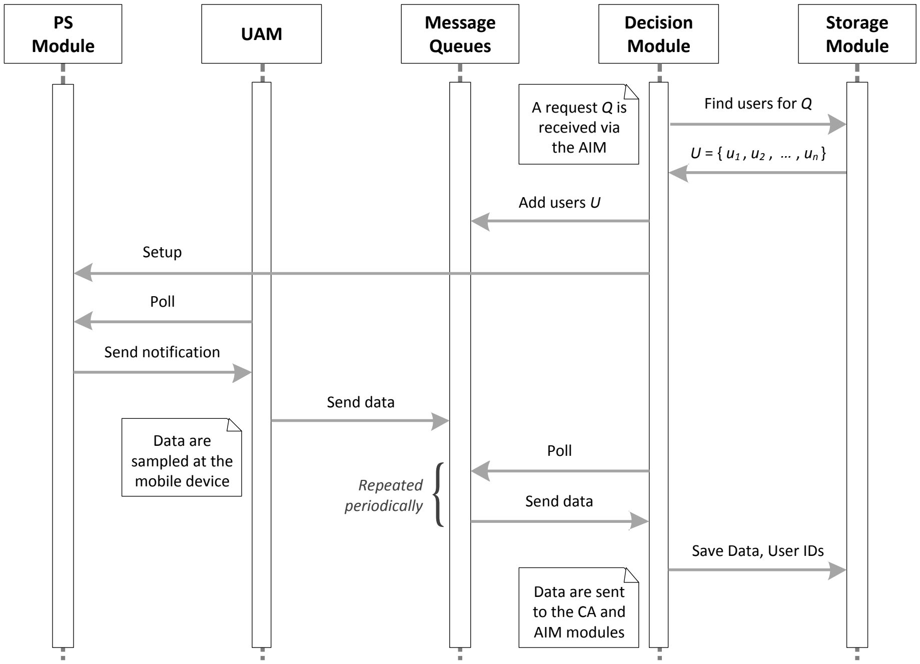

Figure 2: Work flow within the system components for a given request

5.2 Storage Module

In the current implementation, data are divided into scalar and multimedia data (as defined in Section 3) and managed accordingly. In detail, the framework uses two different storage modules: Amazon SimpleDB and Amazon Simple Storage Service (S3). SimpleDB is a non-relational database which provides high availability and simplicity in administration [1]. SimpleDB is used to store scalar (textual) data such as sensory readings. When a scalar data arrives at the MCE, they are associated with the corresponding user ID, which is part of the message sent by the mobile cloud application, and then stored in SimpleDB. The idea behind using a non-relational database is to reduce the database administration overhead and provide fast access to the data [1]. When a mobile phone has multimedia data, the MCA stores them into Amazon Simple Storage Service (S3), and sends a message containing a reference to the data in S3 to SQS. S3 can be used to store and retrieve any amount of data [1]. By using S3 and SimpleDB together, we use the features of SimpleDB as regular database as well as the storage capacity of S3 for multimedia data.

5.3 User Adaptation Module

Data arriving at the UAM are forwarded to the relevant MCE(s). Specifically, when multiple MCEs have common users (i.e., in terms of the corresponding MCAs), the data captured by the MCAs are forwarded to all MCEs. To this end, the UAM implementation is based on two components: a filter process and a queue. The filter process forwards the data to the concerned MCE, while the queue is used by the filter process to forward the data to multiple MCEs.

6. EXPERIMENTAL RESULTS

In order to characterize the advantages of using our framework, we carried out two sets of experiments and studied the impact of the unified sensing approach in terms of energy consumption of the mobile device as well as scalability. In the first set of experiments, we assumed that the proposed framework is deployed in the cloud and is exploited by 4 applications. A thin mobile client installed on the mobile device collectively sampled and uploaded data to the UAM on behalf of the 4 applications. In the second set of experiments, we did not exploit our framework, but rather installed individual applications on the mobile device. We varied the number of applications from 1 to 4. They sampled and uploaded data individually on the mobile device. In all the experiments, data was captured and uploaded to the cloud every second.

The evaluation was conducted by using a Google Nexus One smartphone. The duration of the experiment was set to $1,000$ seconds. We measured the energy consumed by the mobile device for uploading messages in each experiment through the PowerTutor energy profiler application [19]. The obtained results are shown in Figure 3. Each data point represents the average energy consumption over 10 iterations of each experiment, while the error bars represent the related confidence interval with a 95% confidence level. Figure 3 shows that the energy consumption of the mobile devices increase with the number of applications. Even though different applications require the same data, such as GPS coordinates or images, the device separately samples the data for each of them. Hence, as a result, it leads to same data being separately sent to each application. Thus, the mobile device experiences higher energy consumption with the increase in the number of applications. In contrast, the same data required by multiple applications are captured and sent to the UAM only once in our framework. The MCE collects and processes the data from each UAM and forwards it to the relevant application. Thus, this results in a lower energy consumption for the device, as it is not interacting with each application individually.

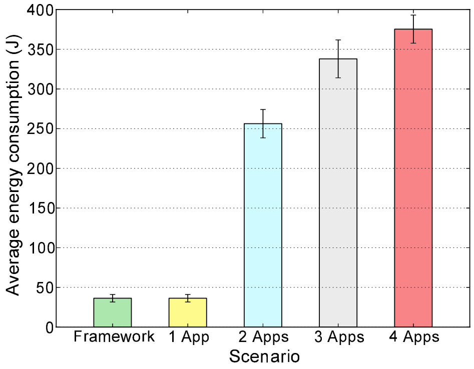

Figure 3: Energy consumption of the mobile phone as a function of the number of applications

Furthermore, we carried out additional experiments to evaluate the scalability and availability of the implemented prototype as a function of the number of users. To emulate users, we deployed 20 virtual machines sending an aggregate message rate ranging from 400 messages/sec to 4×10^5 messages/sec. We found that no messages were delayed at the message queues, thus we can conclude that the prototype is capable of handling a large number of users.

7. CONCLUSION

In this paper, we presented an integrated framework to enable cloud-based and people-centric applications on mobile devices. Our solution exploits the cloud computing paradigm to provide a new delivery model specifically targeted to mobile applications. Through a prototype implementation, we have shown the feasibility of the proposed framework by addressing a case study represented by an emergency response system. Experimental results demonstrate that the proposed approach results in significant energy saving at the mobile device. As a future work, we intend to add more features and functionalities to the proposed framework, with focus on the decision module.

8. ACKNOWLEDGEMENTS

This work has been partially supported by the US National Science Foundation (NSF) under grants CNS-1150192 and IIS-1064460, by the Academy of Finland under grant number 253860, and by an AWS in education research grant. Any opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect those of the NSF and Amazon.

9. REFERENCES

[1] Amazon Inc. Amazon Web Services. `http://aws.amazon.com/`.

[2] A. Campbell, S. Eisenman, N. Lane, E. Miluzzo, R. Peterson, H. Lu, X. Zheng, M. Musolesi, K. Fodor, and G.-S. Ahn. The rise of people-centric sensing. *IEEE Internet Computing*, 12(4):12 –21, July 2008.

[3] CENS/UCLA. Garbage Watch. `http://urban.cens.ucla.edu/projects/gabagewatch/`.

[4] CNN Money. Wildfires and drought cost Texas billions. `http://money.cnn.com/2011/09/08/news/economy/damages_texas_wildfires/index.htm`.

[5] Google Inc. Android 2.1 Platform. `http://developer.android.com`.

[6] Google Inc. Google Fusion Tables. `http://www.google.com/fusiontables`.

[7] M. M. Hassan, B. Song, and E.-N. Huh. A framework of sensor-cloud integration opportunities and challenges. In *Proc. of ICUIMC*, pages 618–626, 2009.

[8] Intel Research. Urban Atmospheres. `http://www.urban-atmospheres.net`.

[9] N. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. Campbell. A survey of mobile phone sensing. *IEEE Communications Magazine*, 48(9):140–150, Sept 2010.

[10] J. Li, Q. Li, S. Khan, and N. Ghani. Community-based cloud for emergency management. In *Proc. of 6th Int'l Conference on System of Systems Engineering*, pages 55 –60, June 2011.

[11] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell. Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application. In *Proc. of SenSys*, pages 337–350, 2008.

[12] M. Mun, S. Reddy, K. Shilton, N. Yau, J. Burke, D. Estrin, M. Hansen, E. Howard, R. West, and P. Boda. Peir, the personal environmental impact report, as a platform for participatory sensing systems research. In *Proc. of MobiSys*, pages 55–68, 2009.

[13] S. Reddy, K. Shilton, G. Denisov, C. Cenizal, D. Estrin, and M. Srivastava. Biketastic: sensing and mapping for better biking. In *Proc. of 28th Int'l conference on Human factors in computing systems*, pages 1817–1820, 2010.

[14] F. Reynolds. Web 2.0–in your hand. *IEEE Pervasive Computing*, 8(1):86 –88, Jan.-March 2009.

[15] J. Rodrigues, A. Aguiar, F. Vieira, J. Barros, and J. Cunha. A mobile sensing architecture for massive urban scanning. In *Proc. of IEEE ITSC*, pages 1132 –1137, Oct. 2011.

[16] The Texas A&M University System. Texas Forest Service. `http://tfsweb.tamu.edu`.

[17] A. Thiagarajan, L. S. Ravindranath, K. LaCurts, S. Toledo, J. Eriksson, S. Madden, and H. Balakrishnan. Vtrack: Accurate, energy-aware traffic delay estimation using mobile phones. In *Proc. of SenSys*, Nov. 2009.

[18] UC Berkeley. Mobile Millennium. `http://traffic.berkeley.edu/`.

[19] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *Proc. of CODES/ISSS '10*, pages 105–114, 2010.

[20] W. Zhang, L. Zhang, Y. Ding, T. Miyaki, D. Gordon, and M. Beigl. Mobile sensing in metropolitan area: Case study in beijing. In *Mobile Sensing Challenges Opportunities and Future Directions, Ubicomp2011 workshop*, 2011.