Mobile Crowdsensing: Current State and Future Challenges

Raghu K. Ganti, Fan Ye, and Hui Lei, IBM T. J. Watson Research Center

ABSTRACT

An emerging category of devices at the edge of the Internet are consumer-centric mobile sensing and computing devices, such as smartphones, music players, and in-vehicle sensors. These devices will fuel the evolution of the Internet of Things as they feed sensor data to the Internet at a societal scale. In this article, we examine a category of applications that we term mobile crowdsensing, where individuals with sensing and computing devices collectively share data and extract information to measure and map phenomena of common interest. We present a brief overview of existing mobile crowdsensing applications, explain their unique characteristics, illustrate various research challenges, and discuss possible solutions. Finally, we argue the need for a unified architecture and envision the requirements it must satisfy.

INTRODUCTION

The integration of sensing and embedded everyday computing devices at the edge of the Internet will result in the evolution of an *embedded Internet* or the *Internet of Things* (IoT). Typical IoT devices include physical items tagged/embedded with sensors (e.g., chemical containers with temperature sensors), scissors with integrated circuit (IC) tags, and smart meters to remotely monitor energy consumption. An emerging category of edge devices that we believe will result in the evolution of the IoT are consumer-centric mobile sensing and computing devices, which are connected to the Internet. These include smartphones (iPhone, Google Nexus), music players (iPods), sensor embedded gaming systems (Wii, XboX Kinect), and in-vehicle sensing devices (GPS, OBD-II). They have become extremely popular recently and are potentially important sources of sensor data. They are typically equipped with various sensing faculties and wireless capabilities that allow them to produce data and upload the data to the Internet. As an example, a sample list of mobile devices and their corresponding sensing capabilities are provided in Table 1. Future sensing capabilities on smartphones include ECG (for medical purposes, e.g., ithlete, H'andy Sana), poisonous chemical detection (e.g., cell-all), and air quality sensors (e.g., Intel's EPIC, Fig. 1).

Different from the "typical" everyday IoT objects (e.g., coffee machines) that traditionally lack computing capabilities, these mobile devices have a variety of sensing, computing, and communication faculties. They can either serve as a bridge to other everyday objects, or generate information about the environment themselves. We believe they will drive a plethora of IoT applications that elaborate our knowledge of the physical world.

These applications can be broadly classified into two categories, *personal* and *community* sensing, based on the type of phenomena being monitored. In personal sensing applications, the phenomena pertain to an individual; for example, the monitoring of movement patterns (e.g., running, walking, exercising) of an individual for personal record-keeping or healthcare reasons. Another example of personal sensing is one that monitors the transportation modes of an individual to determine his or her carbon footprint.

On the other hand, community sensing pertains to the monitoring of large-scale phenomena that cannot easily be measured by a single individual. For example, intelligent transportation systems may require traffic congestion monitoring and air pollution level monitoring. These phenomena can be measured accurately only when many individuals provide speed and air quality information from their daily commutes, which are then aggregated spatio-temporally to determine congestion and pollution levels in cities.

Community sensing is also popularly called *participatory* sensing [1] or *opportunistic* sensing [2]. Participatory sensing requires the active involvement of individuals to contribute sensor data (e.g., taking a picture, reporting a road closure) related to a large-scale phenomenon. Opportunistic sensing is more autonomous, and user involvement is minimal (e.g., continuous location sampling without the explicit action of the user). We take the position that community sensing spans a wide spectrum of user involvement, with participatory sensing and opportunistic sensing at the two ends. We therefore coin the term *mobile crowdsensing* (MCS) to refer to a broad range of community sensing paradigms.[1]

In the rest of this article, we survey existing crowdsensing (both participatory and opportunistic) applications, identify unique characteristics of MCS applications, and discuss the

[1] *The notion that crowdsensing spans a spectrum from participatory to opportunistic sensing was suggested by our colleague Thomas Erickson.*

 0163-6804/11/$25.00 © 2011 IEEE

Device	Inertial	Compass	GPS	Microphone	Camera	Proximity	Light
iPhone 4	✓	✓	✓	✓	✓	✓	✓
Nexus S	✓	✓	✓	✓	✓	✓	✓
Galaxy S II	✓	✓	✓	✓	✓	✓	✓
HTC Sensation	✓	✓	✓	✓	✓	✓	✓
Garmin ForeRunner 410	✓	✓	✓				

Table 1. *Sensors on various mobile sensing devices.*

In environmental MCS applications, the phenomena are those of the natural environment. Examples include measuring pollution levels in a city, water levels in creeks, and monitoring wildlife habitats. Such applications enable the mapping of various large-scale environmental phenomena by involving the common person.

Figure 1. *Sensors on current iPhone 4 and future sensors that will possibly be integrated with mobile phones: a) sensors on Apple's iPhone 4; b) Intel's air quality sensor that communicates with Bluetooth enabled mobile phones; c) ECG sensor enabled mobile phone, H'andy Sana.*

research challenges they face, their solutions and trade-offs. The research challenges we discuss include *localized analytics*; *resource limitations*; *privacy*, *security, and data integrity*; *aggregate analytics*; and *architecture*.

Mobile CrowdSensing Applications

In this section, we briefly discuss existing mobile crowdsensing applications, which provide a basis for illustrating various research challenges in the rest of this article. We classify MCS applications into three categories based on the type of phenomenon being measured or mapped. These include *environmental*, *infrastructure*, and *social*.

In environmental MCS applications, the phenomena are those of the natural environment. Examples include measuring pollution levels in a city, water levels in creeks, and monitoring wildlife habitats. Such applications enable the mapping of various large-scale environmental phenomena by involving the common person. An example prototype deployment for pollution monitoring is Common Sense [3]. Common Sense uses specialized handheld air quality sensing devices that communicate with mobile phones (using Bluetooth) to measure various air pollutants (e.g., CO_2, NO_x). These devices, when deployed across a large population, collectively measure the air quality of a community or a large area. Similarly, one can utilize microphones on mobile phones to monitor noise levels in communities. Another example is *CreekWatch* developed by IBM Almaden Research Center. It monitors water levels and quality in creeks by aggregating reports from individuals, such as pictures taken at various locations along the creek or text messages about the amount of trash. Such information can be used by water control boards to track pollution levels in water resources.

Infrastructure applications involve the measurement of large-scale phenomena related to public infrastructure. Examples include measuring traffic congestion, road conditions, parking availability, outages of public works (e.g., malfunctioning fire hydrants, broken traffic lights), and real-time transit tracking. Early MCS deployments measured traffic congestion levels in cities, examples of which include MIT's CarTel [4] and Microsoft Research's Nericell [5]. CarTel utilizes specialized devices installed in cars to measure the location and speed of cars, and transmit the measured values using public WiFi hotspots to a central server. This central server can then be queried to provide information such as least delay routes or traffic hotspots. On the other hand, Nericell utilizes individuals' mobile phones to not only determine average speed or traffic delays, but also detect honking levels (especially in countries like India where honking is common) and potholes on roads. Another example is ParkNet [6], an application that detects available parking spots in cities using ultrasonic sensing devices installed on cars combined with smart phones.

Finally, the third category is social applications, where individuals share sensed informa-

To efficiently support multiple concurrent applications, it is critical to identify common data needs and support the reuse of sensor data across applications. In contrast, a conventional sensor network is typically intended for a single application, and reuse for vastly different purposes is rarely needed.

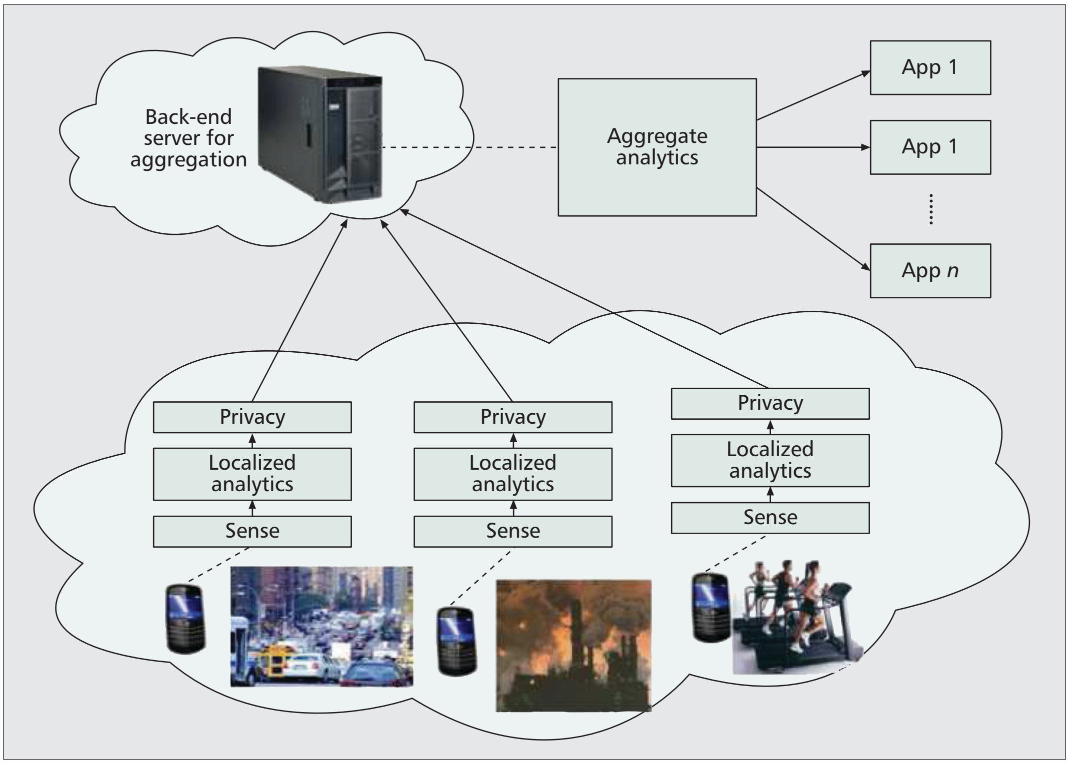

Figure 2. *Typical functioning of MCS applications. Raw sensor data are collected on devices and processed by local analytic algorithms to produce consumable data for applications. The data may then be modified to preserve privacy and is sent to the backend for aggregation and mining.*

tion among themselves. As an example, individuals can share their exercise data (e.g., how much time one exercises in a day) and compare their exercise levels with the rest of the community. They can use this comparison to help improve their daily exercise routines. Example deployments include BikeNet [7] and DietSense [8]. In BikeNet, individuals measure location and bike route quality (e.g., CO_2 content on route, bumpiness of ride), and aggregate the data to obtain the "most" bikeable routes. In DietSense, individuals take pictures of what they eat and share it within a community to compare their eating habits. A typical use case for this is for a community of diabetics to watch what other diabetics eat and control their diet or provide suggestions to others.

To summarize, the functioning of typical MCS applications is illustrated in Fig. 2, which depicts a number of research challenges as functional components.

MCS: Unique Characteristics

We first illustrate the unique characteristics of MCS applications that differentiate them from traditional mote-class sensor networks. This provides the reader with an idea of the research challenges faced by MCS applications.

Compared to traditional mote-class sensor networks, MCS has a number of unique characteristics that bring both new opportunities and problems. First, today's mobile devices have significantly more computing, communication, and storage resources than mote-class sensors, and are usually equipped with multimodality sensing capabilities. These will enable many applications that require resources and sensing modalities beyond those current mote-class sensors possess. Second, millions of mobile devices are already "deployed in the field": people carry these devices wherever they go and whatever they do. By leveraging these devices, we could potentially build large-scale sensing applications efficiently (cost and time). For example, instead of installing roadside cameras and loop detectors, we can collect traffic data and detect congestion levels using smartphones carried by drivers. Such solutions reduce the cost of deployment of specialized sensing infrastructure.

The dynamic conditions of the set of mobile devices and the need for data reuse across different applications in MCS are also quite different from those of traditional sensor networks. In MCS, the population of mobile devices, the type of sensor data each can produce, and the quality in terms of accuracy, latency, and confidence can change all the time due to device mobility, variations in their energy levels and communication channels, and device owners' preferences. Identifying the right set of devices to produce the desired data and instructing them to sense with proper parameters to ensure the desired quality is a complex problem. In traditional sensor networks, the population and the data they can produce are mostly known a priori; thus, controlling the data quality is much easier. The same sensor data have been used for different purposes in many existing MCS applications. For example, accelerometer readings have found usage in transportation mode identification, pothole detection, and human activity pattern extraction.

To efficiently support multiple concurrent applications, it is critical to identify common data needs and support the reuse of sensor data across applications. In contrast, a conventional sensor network is typically intended for a single application, and reuse for vastly different purposes is rarely needed.

Because devices are owned and carried by individual users, humans are usually involved in the loop. On one hand, the intelligence and mobility of humans can be leveraged to help applications collect higher-quality or semantically complex data that might otherwise require sophisticated hardware and software. For example, humans can easily identify available street parking spots and report with pictures or text messages, whereas an ultrasound-based scanning system not only requires special hardware, but also sophisticated processing algorithms to ensure the reliability of data. On the other hand, humans naturally have privacy concerns and personal preferences that are not necessarily aligned with the end goals of the MCS applications. The user may not want to share sensor data that contains or reveals private and sensitive information, such as their current location.

Another important implication for human involvement is incentive. Participating individuals (devices) may incur energy and monetary costs, or even explicit efforts by the owner of the device for sensing, processing, and communicating desired data. Unless there are strong enough incentives, the owners may not be willing to contribute their resources. For MCS applications to succeed, there have to be appropriate incentive mechanisms to recruit, engage, and retain human participants. Elaboration on incentive mechanisms and other people-oriented tools is beyond the scope of this article, since our focus is on system challenges.

Localized Analytics

Various sensors such as GPS, accelerometer, microphone and camera are available on mobile devices. The operating ssytem (OS) allows applications to access the sensors and extract raw sensing data from them. However, depending on the nature of the raw data and the needs of applications, the physical readings from sensors may not be suitable for the direct consumption of applications. Many times, some *local analytics* performing certain primitive processing of the raw data on the device are needed. They produce intermediate results, which are sent to the back-end for further processing and consumption. For example, in a pothole detection [5] application, a local analytic computes spikes from 3-axis acceleration sensor data to determine potential potholes.

The motivation of such local analytics are twofold. First, the kind of processing performed leads to appropriately summarized data, thus consuming less energy and bandwidth than transmitting the raw sensor readings. This is a well-known trade-off in conventional mote-class sensor networks: using computation to save energy/bandwidth. Second, it reduces the amount of processing that the back-end has to perform. Furthermore, if the mobile devices in a societal-scale deployment transmit raw sensor data, the back-end can easily be overwhelmed. Finally, some applications are delay sensitive, and transmitting raw sensor data on intermittently connected channels can be more time consuming than sending processed sensor data.

The main challenge in local analytics is finding heuristics and designing algorithms to achieve the desired function. One category of functions is *data mediation*, such as filtering of outliers, elimination of noise, or filling in data gaps. For example, GPS samples acquired may not be accurate or missing (due to lack of line of sight), in which event outliers need to be eliminated or missing samples extrapolated.

Another common category of functions is *context inference*. Examples of context include transportation mode (whether the user is on a car, bus, train, or on foot), the kinetic modes of humans (walking, standing, jogging, running), the social settings (e.g., in a meeting, on a phone call, watching TV), or the occurrence of certain events in the surrounding environments (e.g., potholes on the road, stop-and-go traffic, loud noise levels). The heuristics and algorithms needed can be quite application-specific. Hence, the exact algorithm used for context inference depend on the nature of the application and the characteristics of the context. The current practice is to develop analytics solely for one application. This could lead to an "explosion" of analytics when many crowdsensing applications coexist. Each analytic is working individually, and there is a possibility that they may access the same sensor or involve similar computation in their inference.

Resource Limitations: Energy, Bandwidth, and Computation

Even though they possess much more computing, bandwidth, and energy resources than mote-class sensors, mobile devices nevertheless face resource limitations. Resource constraints in traditional sensor networks have been well studied. However, MCS applications introduce new aspects in this regard.

First, the set of devices that are collecting sensor data are highly dynamic in availability and capabilities. Due to this highly dynamic nature, modeling and predicting the energy and bandwidth requirements to accomplish a particular task is harder than traditional sensor networks. Second, when there are a large number of available devices with diverse sensing capabilities, identifying and scheduling sensing and communication tasks among them under resource constraints are more complex.

Another interesting aspect is the interdependencies between various types of sensory data due to multimodality sensing capabilities. Different types of data can be used for the same purpose, but with different quality and resource consumption trade-offs. Leveraging these differences to improve the quality while minimizing resource consumption is a novel challenge. For example, location data can be provided using GPS, WiFi, and GSM, with decreasing levels of accuracy. Compared to WiFi and GSM, continu-

Even though they possess much more computing, bandwidth, and energy resources than mote-class sensors, mobile devices nevertheless face resource limitations. Resource constraints in traditional sensor networks have been well studied. However, MCS applications introduce new aspects in this regard.

Applications performing sensing and processing activities independently without understanding the consequences on each other will result in low efficiency on an already resource-constrained platform. There is a high likelihood of duplicating sensing and processing across multiple applications.

ous GPS location sampling drains the battery faster. One approach to this problem uses low duty cycling to reduce energy consumption of high-quality sensors (i.e., GPS), and alternates between high- and low-quality sensors depending on the energy levels of the device (e.g., sample WiFi often when battery level is less than 70 percent). This approach trades off data quality and accuracy for energy.

The existence of multiple concurrent applications that require data of different types also complicates resource allocation. A mobile device can be sampling various sensors (e.g., GPS, accelerometer, air quality) on behalf of different applications. The approach proposed by CarTel prioritizes data collection tasks. Depending on the priority of the application that requires sensor data, the sampling rate of other sensors can be reduced (or the sensor completely switched off). For example, during peak travel times, a community may be more interested in obtaining traffic congestion levels as opposed to air or noise pollution levels. As a result, the air/noise sensors sample much less frequently or can be shut down.

A drawback of existing solutions such as low duty cycling are that they are designed for their particular application context and do not scale when many different applications coexist. An important challenge for large-scale deployment of MCS applications is that the resource constraints need to be addressed in a holistic manner. How do multiple applications on the same device utilize energy, bandwidth, and computation resources without significantly affecting the data quality of each other? How does scheduling of sensing tasks occur across multiple devices with diverse sensing capabilities and availabilities (which can change dynamically)? We believe that these questions need to be answered before MCS applications can be deployed on a large scale.

Privacy, Security, and Data Integrity

An important aspect of MCS applications is that they potentially collect sensitive sensor data pertaining to individuals. For example, GPS sensor readings can be utilized to infer private information about the individual, such as the routes they take during their daily commutes,and their home and work locations [9]. On the other hand, these GPS sensor measurements (from daily commutes) shared within a larger community can be used to obtain traffic congestion levels in a given city [4]. Thus, it is important to preserve the security and privacy of an individual, but at the same time enable MCS applications. It is also necessary to ensure that an individual's sensor data is not revealed to untrustworthy third parties. A problem that arises from the opt-in nature of crowdsensing applications is when malicious individuals contribute erroneous sensor data (e.g., falsified GPS readings); hence, maintaining the integrity of sensor data collected is an important problem. In what follows, we briefly touch on these challenges.

A popular approach to preserving privacy of data is anonymization [10], which removes any identifying information from the sensor data before sharing it with a third party. The drawback of such an approach is that anonymized GPS (or location) sensor measurements can still be used to infer the frequently visited locations of an individual and derive their personal details. Another approach to preserving privacy is secure multiparty computation [11], where cryptographic techniques are used to transform the data in order to preserve privacy. Such cryptographic techniques are compute intensive and not scalable because they require the generation and maintenance of multiple keys, which also leads to higher energy consumption. We believe that *data perturbation* based approaches, which add noise to sensor data before sharing it with the community to preserve the privacy of an individual, are appropriate. Data perturbation approaches [12, 13] rely on adding noise in such a manner that the privacy of an individual is preserved, but at the same time it is possible to compute the statistics of interest with high accuracy (due to the nature of the noise being added). For example, in a weight watchers application, it is important to compute the average weight of the population. Each individual i is sensitive to revealing his or her weight (w_i) and perturbs it by adding a random number (r_i), which is drawn from a known distribution with zero mean. Although individual weights are perturbed and appear random, when these values are averaged, the randomized component ($\sum r_i$) vanishes (given a sufficient number of individuals), and the average weight of the community can be computed with a high degree of accuracy.

Data integrity, ensuring the integrity of sensor data generated by individuals, needs to be addressed by MCS applications. Some approaches have been proposed in existing literature [14, 15], which rely on collocated infrastructure as a *witness*. Such an approach relies on inputs from the installation of expensive infrastructure, which can be prohibitive and unavailable at times. Another approach is to *sign* the sensor data (by trusted hardware installed on mobile phones). However, this approach is potentially problematic as the verification has to be done even in the software. For example, a GPS location may be perturbed by adding noise for privacy reasons or audio sensor data may be processed to conserve energy.

We make a few observations in regard to privacy, security, and data integrity for MCS applications. First, we observe that privacy is very user specific, that is each individual has a different perception of privacy. For example, one person may be willing to share his or her location information continuously, whereas another may not. Developing privacy techniques that address variation in individual preferences is needed. Furthermore, a generic perturbation technique or a framework of perturbation techniques need to be developed such that privacy and security can be achieved in a generic setting independent of the nature of the data being shared. Finally, real-world MCS deployments must address the data integrity problem in order to provide meaningful conclusions from the aggregate sensor data.

Aggregate Analytics

The local analytics running on mobile devices only analyze data on the given device. MCS applications rely on analyzing the data from a collection of mobile devices, identifying spatio-temporal patterns. For example, the transportation authority of a city may be interested in the spatial distribution of traffic hot spots around the road network, and how the distribution evolves over various time scales. Such insight can help them better coordinate the traffic lights to ease traffic depending on the time of the day, and better plan future road expansions in the long term to reduce congestion. Another example is for public works maintenance. Citizens can report problems in public facilities, such as broken water pipes and dysfunctional traffic lights. Such reports can be used by maintenance personnel to infer (to a certain degree) the impact and severity of the incident to help prioritize and schedule the repair resources.

The patterns may also help users build models and make predictions about the physical or social phenomena being observed. One example is the monitoring of pollutants such as car exhaust. An important aspect of environment protection is to build models to understand the dissemination of pollutants in the air, soil and water. By collecting large amount of data samples about air pollutants such as car exhaust, one can not only monitor the concentration of pollution, but also detect patterns to model how the concentration evolves spatially and temporally as temperature, humidity and wind change. These models can help the environmental authority forecast and provide alerts to the public.

The challenge in identifying patterns from large amounts of data is usually application-specific and involves certain data mining algorithms. Depending on the amount of incoming data and the delay sensitivity of applications, there are two possible approaches for data mining. One is a traditional approach where data is stored in a database first, and then one can apply various mining algorithms against the database to detect patterns. However, if the amount of continuous data input is too much for storage, or the application requires fast detection of patterns, stream data mining algorithms may be required. Such algorithms take as input continuous data streams and identify patterns, without the need to first store the data. Data mining algorithms are domain-specific, and the exact algorithms will be closely related to the application and are out of scope of this article.

Architecture

In this section, we illustrate the current state of the architecture of existing MCS applications and point out its drawbacks. Currently, a typical MCS application has two application specific components, one on the device (for sensor data collection and propagation) and the second in the backend (or cloud) for the analysis of the sensor data to drive the MCS application. This architecture is depicted in Fig. 3. We refer to this as *application silos* because each application is built ground-up and independent from each other. There is no common component even though each application faces a number of common challenges in data collection, resource allocation and energy conservation.

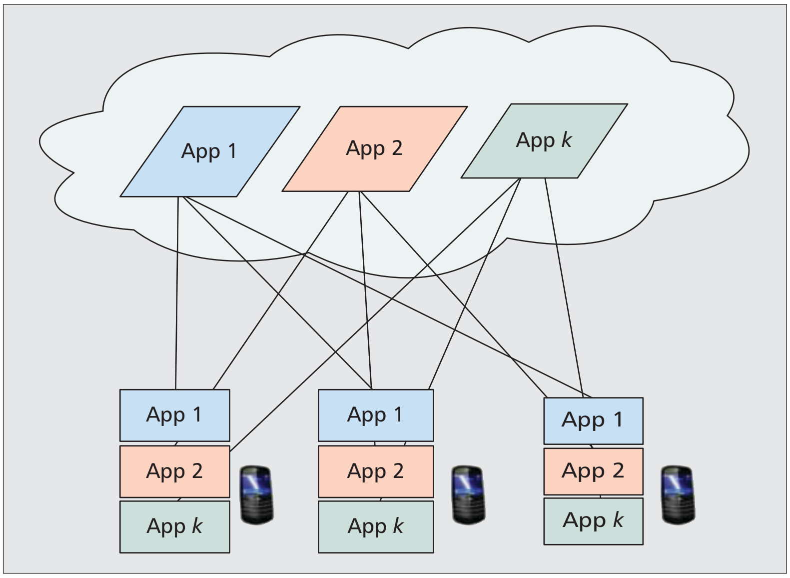

Figure 3. *Existing MCS applications take an "application silo" approach where each application is built from scratch without any common component even though they face many common challenges. Such an architecture hinders the development of new MCS applications and we envision a unifying architecture should address its limitations.*

Such an architecture hinders the development and deployment of MCS applications in several ways. First, it is hard to *program* an application. To write a new application, the developer has to address challenges in energy, privacy, and data quality in an ad hoc manner, reinventing the wheel all the time. Further, he may need to develop different variants of local analytics if he wants to run the application on heterogeneous devices using different OSes. Second, this approach is *inefficient*. Applications performing sensing and processing activities independently without understanding the consequences on each other will result in low efficiency on an already resource constrained platform. There is a high likelihood of duplicating sensing and processing across multiple applications. For example, traffic sensing, air and noise pollution all require location information, but these applications would each do its own sampling without reusing the same data samples. Furthermore, there is no collaboration or coordination across devices. Devices may not all be needed (e.g., traffic sensing in a given location), especially when the device population is dense. Finally, the current architecture is not *scalable*. Only a small number of applications can be accommodated on each device (e.g., limitations imposed by the device operating system, human capacity to keep track of a large number of applications). Also, the data gathered from societal-scale sensing may overwhelm network and back-end server capacities, thus making the current architecture non-scalable.

We envision that a unifying architecture could address the current limitations of how

Further Reading

As such, extensive research has been conducted in each of the above addressed problems, and we provide some further readings in this breakout box.

Localized Analytics

- A. Thiagarajan *et al.*, "Cooperative Transit Tracking Using GPS-Enabled Smart-Phones," *Proc. SenSys 2010*, pp. 85–98.
- E. Miluzzo *et al.*, "Sensing Meets Mobile Social Networks: The Design, Implementation, and Evaluation of the CenceMe Application," *Proc. SenSys 2008*, pp. 337–50.

Resource Limitations

- A. Thiagarajan *et al.*, "Accurate, Low-Energy Trajectory Mapping for Mobile Devices," *Proc. NSDI 2011*.
- Moo-Ryong Ra *et al.*, "Energy-Delay Tradeoffs in Smartphone Applications," *Proc. MobiSys 2010*, pp. 255–70.
- A. Balasubramanian, R. Mahajan, and A. Venkataramani, "Augmenting Mobile 3G Using WiFi: Measurement, Design, and Implementation," *Proc. MobiSys 2010*, pp. 209–22.

Privacy, Security, and Data Integrity

- S. Saroiu and A. Wolman, "I Am A Sensor, and I Approve This Message," *Proc. HotMobile 2010*, pp. 37–42.
- P. Gilbert *et al.*, "Toward Trustworthy Mobile Sensing," *Proc. HotMobile 2010*, pp. 31–36.
- A. Kapadia, D. Kotz, and N. Triandopoulos, "Opportunistic Sensing: Security Challenges for the New Paradigm," *Proc. COMSNETS 2009*, pp. 127–36.

Aggregate Analytics

- Z. Li *et al.*, "MoveMine: Mining Moving Object Data for Discovery of Animal Movement Patterns," *ACM Trans. Intelligent Systems and Technology*, Aug. 2010.
- M. Demirbas *et al.*, "iMap: Indirect Measurement of Air Pollution with Cellphones," *Proc. PerCom*, 2009, pp. 1–6.

Architecture

- D. Trossen and D. Pavel, "NORS: An Open Source Platform to Facilitate Participatory Sensing with Mobile Phones," *Proc. MobiQuitous 2007*, pp. 1–8.
- M. Mun *et al.*, "PEIR, the Personal Environmental Impact Report, As A Platform for Participatory Sensing Systems Research," *Proc. MobiSys 2009*, pp. 55–68.
- B. Demchak *et al.*, "A Rich Services Approach to CoCoME," *LNCS*, vol. 5153, Aug. 2008, pp. 85–115.

MCS applications are developed and deployed. It will satisfy the common needs for multiple different applications. First, it should allow application developers to specify their data needs in a high-level language. It should identify common data needs across applications to avoid duplicate sensing and processing activities on devices. Second, it should automatically identify the set of devices that can provide the desired data, and produce instructions to configure the sensing activities on devices properly. When dynamic changes happen, it should adapt the set of chosen devices and sensing instructions to ensure the desired data quality. Finally, to avoid writing different versions of local analytics on heterogeneous devices, a layer that can shield the differences in physical sensor access application programming interfaces (APIs) and provide the same API upward is necessary. This makes it possible to reuse the same local analytics across different device platforms, assuming these platforms all support a common programming language such as Java.

Conclusions and Future Work

In conclusion, we have identified a category of IoT applications that rely on data collection from large number of mobile sensing devices such as smartphones, which we termed mobile crowdsensing (MCS). We have presented several MCS applications, such as CarTel, Nericell, ParkNet, BikeNet, and DietSense. We have identified the unique characteristics of MCS, presented several research challenges of MCS, and discussed their solutions briefly. We also note that due to space limitations, we have not presented all of the existing work (in terms of applications as well as the individual research challenges). We are currently exploring a unified architecture for collecting and processing sensor data from mobile sensing devices at a societal scale.

Acknowledgments

We would like to thank Joel Branch, Han Chen, Sastry Duri, Minkyong Kim, Johnathan Reason, and Zhe Zhang for several stimulating discussions that helped shape this article. This article was initially written when we were defining an Open Collaborative Research project between IBM Research and universities. We would like to acknowledge our academic collaborators Tarek Abdelzaher, Jiawei Han, Tian He, and Loren Terveen, as well as other IBM researchers on the project — Maria Ebling, Thomas Erickson, and Wendy Kellogg.

References

[1] J. Burke *et al.*, "Participatory Sensing," *Wksp. World-Sensor-Web, Collocated with ACM SenSys*, 2006; http://www.sensorplanet.org/wsw2006/.

[2] N. Lane *et al.*, "A Survey of Mobile Phone Sensing," *IEEE Commun. Mag.*, vol. 48, no. 9, 2010, pp. 140–50.

[3] P. Dutta *et al.*, "Demo Abstract: Common Sense: Participatory Urban Sensing Using A Network of Handheld Air Quality Monitors," *Proc. ACM SenSys*, 2009, pp. 349–50.

[4] B. Hull *et al.*, "Cartel: A Distributed Mobile Sensor Computing System," *Proc. SenSys*, 2006, pp. 125–38.

[5] P. Mohan, V. Padmanabhan, and R. Ramjee, "Nericell: RICH monitoring of Road and Traffic Conditions Using Mobile Smartphones," *Proc. ACM SenSys*, 2008, pp. 323–36.

[6] S. Mathur *et al.*, "Parknet: Drive-by Sensing of Road-Side Parking Statistics," *Proc. ACM MobiSys*, 2010, pp. 123–36.

[7] S. B. Eisenman *et al.*, "The Bikenet Mobile Sensing System for Cyclist Experience Mapping," *Proc. SenSys*, Nov. 2007.

[8] S. Reddy *et al.*, "Image Browsing, Processing, and Clustering for Participatory Sensing: Lessons from a Dietsense Prototype," *Proc. EmNets*, 2007, pp. 13–17.

[9] J. Krumm, "A Survey of Computational Location Privacy," *Personal and Ubiquitous Computing*, vol. 13, no. 6, 2009, pp. 391–99.

[10] L. Sweeney, "k-Anonymity: A Model for Protecting Privacy," *Int'l. J. Uncertainty, Fuzziness, and Knowledge-Based Systems*, vol. 10, no. 5, Oct. 2002, pp. 557–70.

[11] A. Yao, "Protocols for Secure Computations," *Proc. IEEE Symp. Foundations of Comp. Sci.*, 1982, pp. 160–64.

[12] R. Agrawal and R. Srikant, "Privacy Preserving Data Mining," *Proc. ACM Conf. Mgmt. of Data*, May 2000, pp. 439–50.

[13] R. K. Ganti *et al.*, "Poolview: Stream Privacy for Grassroots Participatory Sensing," *Proc. SenSys '08*, 2008, pp. 281–94.

[14] V. Lenders *et al.*, "Location-based Trust for Mobile User-Generated Content: Applications, Challenges, and Implementations," *Proc. HotMobile*, 2008, pp. 60–64.

[15] S. Saroiu and A. Wolman, "Enabling New Mobile Applications With Location Proofs," *Proc. HotMobile*, 2009, pp. 1–6.

Biographies

RAGHU GANTI (rganti@us.ibm.com) is a research staff member at the IBM T. J. Watson Research Center. He is part of the IT and Wireless Convergence Department. His research interests span wireless sensor networks, privacy, and data mining. He obtained his M.S. and Ph.D. degrees from the Department of Computer Science, University of Illinois, Urbana-Champaign in August 2010. He is the recipient of the Siebel Scholar fellowship, Class of 2010, which is awarded to the top 80 graduate students across the world's leading Computer Science Departments. He received his B.Tech degree from the Indian Institute of Technology, Madras in computer science and engineering.

FAN YE (fanye@us.ibm.com) is a research staff member in the Policy Technologies group at IBM T. J. Watson Research Center. His previous work includes cloud messaging, wide-area publish/subscribe systems, distributed stream processing systems, and mobile and wireless sensor networking. He is currently working on scalable information collection from mobile devices. He received his Ph.D. in computer science in 2004 from the University of California at Los Angeles, and his M.S. and B.E. from Tsinghua University.

HUI LEI (hlei@us.ibm.com) is a research staff member at the IBM T. J. Watson Research Center, where he manages the Cloud Management Services Department. His research interests include mobile and pervasive computing, cloud computing, and enterprise computing, with an emphasis on software infrastructure and data management issues. He received a Ph.D. in computer science from Columbia University.