# Integrating Participatory Sensing in Application Development Practices

Jamie Payton
Department of Computer Science
The University of North Carolina at Charlotte
payton@uncc.edu

Christine Julien
The EDGE Center
The University of Texas at Austin
c.julien@mail.utexas.edu

## ABSTRACT

With the widespread capabilities of commodity mobile devices, applications will increasingly incorporate *participatory sensing* functionality. Participatory sensing directly involves end-users in collecting (and ultimately sharing) information about the environment. Applications that rely on participatory sensing range from those that simply enable information sharing, to environmental monitoring and response, and route and behavior planning. As more and more applications demand the incorporation of participatory sensing, it becomes imperative to create software architectures, design patterns, and programming libraries that enable the integration of participatory sensing with software engineering theory and practice. In this position paper, we explore the new challenges that participatory sensing applications present, specifically focusing on challenges that demand a reevaluation of software engineering design principles, tools, and techniques. For these challenges, we also posit possible ways forward.

## Categories and Subject Descriptors

D.2.10 [**Software Engineering**]: Design; D.2.2 [**Software Engineering**]: Design Tools and Techniques

## General Terms

Design, Theory

## Keywords

participatory sensing, design patterns, software architecture, programming libraries, techniques, tools

## 1. INTRODUCTION

Participatory sensing applications rely on the participation of end users with mobile computing devices (usually smartphones) to create interactive sensor networks that enable data gathering, analysis, and sharing. In the past five years, participatory sensing applications have been developed in domains ranging from pure information sharing [6, 22], to participatory environmental monitoring [2, 17, 21], to social network applications [9, 11] and route and behavior planning [16]. From these wildly successful initial forays, it is evident that participatory sensing will be a key component of many emerging software systems. However, while the need for more generalized participatory sensing programming models has been previously motivated [3], existing applications have been developed largely in isolation, as siloed approaches. There is clearly a need to bring more focus to the underpinnings of developing participatory sensing applications and the ensuing software engineering challenges. *This will require new research in software architectures, design patterns, programming libraries, and tools that support participatory sensing.*

Simply, participatory sensing involves determining the state of some aspect of the environment using information sensed from volunteers and their computing devices. Most simply, imagine all cell phone users volunteering to have their position on the nation's highway system periodically reported to a database; querying this database would enable an application to estimate traffic conditions based on the densities of reported locations. In this paper, we refer to the collection of participants contributing information as the *crowdsource*. To date, participatory sensing has generated significant buzz, and the applications described above are intuitive and easy to use. However, participatory sensing has not yet been investigated from a design perspective, particularly with respect to a rigorous evaluation of the necessary components that go into the design of participatory sensing applications. Such an undertaking is a necessary first step in enabling the robust, reliable, trustworthy development of participatory sensing applications. We investigate the challenges associated with this development in four pieces:

- *Defining the Crowdsource.* Recent work has investigated developing a *recruitment framework* for identifying likely candidates for the crowdsource and requesting their involvement [14]. To enable large-scale incorporation of participatory sensing into software engineering tools and methods requires abstractions that enable application developers to provide specifications of the best contributing participants (in terms of locality, temporality, and responsiveness) and to acquire their participation.
- *Maintaining the Privacy of the Crowdsource.* As participants give up information collected by their personal mobile devices, privacy becomes an essential is-

sue. This information gives away personal information about the participant and may be accessed by a wide variety of applications and other users. This need for privacy pervades all of the other challenges we identify here. While existing work has elucidated the challenges [18] and some initial approaches to providing privacy through aggregation [5]. Any new tools and techniques for engineering participatory sensing software must consider privacy as an essential component.

- *Evaluating the Fidelity of Crowdsourced Information.* When individual participants are relied on to provide data in participatory sensing applications, understanding the quality, or *fidelity* of the acquired information is essential. This has been addressed for purposed sensing [12]. In participatory sensing, existing work uses active feedback mechanisms to assess data quality [13]. To integrate participatory sensing into our mainstream applications, our software engineering tools must enable reasoning about the quality of crowdsourced data.

- *Integrating Participatory Sensing with Purposed Sensing.* While our focus is on participatory sensing, much effort has already been extended with respect to building purposed sensing systems. With the increase in participatory sensing applications, it becomes difficult to make design decisions that weigh participatory sensing against more traditional purposed sensing [8]. Tools must be created that support these decision processes and enable the mixing of the two sensing modalities.

In this paper, we explore these challenges and motivate the need for a careful examination of a support infrastructure for participatory sensing; such an infrastructure is needed to bring participatory sensing applications to the mainstream.

## 2. CHALLENGES AND DIRECTIONS

The software engineering community must put forth a principled approach to participatory sensing to meet demand for rapid and dependable development of future applications. In this section, we explore the four categories of research challenges we have identified that must be addressed to develop a framework that supports participatory sensing.

### 2.1 Defining the Crowdsource

A participatory sensing application is only as good as the participants that support it. In the design of a participatory sensing application, the developer must identify the kind of information required and the best set of participants to provide that information. To support the development of applications, therefore, we need tools and techniques that can support discovery, recruitment, and maintenance of a set of participants. In selecting the participants to support a particular application, we must be able to consider and constrain the participants based on:

- their reputation and trustworthiness;
- both their temporal and geographic availability;
- their ability to cover the sensed phenomena; and
- the quality of their information contributions.

Existing work has looked at using mobility profiles to estimate the potential temporal and geographic coverage of a selected set of participants [15] and at evaluating the reputation of potential participants [14]. To make the integration of participatory sensing into applications a mainstream task, we must generalize, standardize, and ease the process of identifying, selecting, recruiting, and maintaining a meaningful set of participants.

At the same time, the degree of participation required can have a significant impact on a participant's willingness to contribute their time and their devices to the cause. There is effectively a spectrum that ranges from *active* participatory sensing, in which the participants are completely involved in the process, to the point of generating data reports (as in Project BudBurst [2]), to completely *passive* participatory sensing, where, once the participants agree to participate, they can ignore the application entirely, while it functions in the background (as in Nericell [10] or the Pothole Patrol [4]).

To handle these challenges, we need a set of rigorous approaches to handling the discovery and recruitment of participants for participatory sensing application components. We believe this requires a three-pronged approach. First is the need to standardize the protocols used to request participation and maintain connections to participants to retrieve data. This effectively is a complex connector that connects the application with the participants in a software architecture for participatory sensing. Second is the need to build specification languages and libraries that enable applications to expressively specify the nature of the applications' ideal participants based on the axes identified above. Such specification languages should be easy to use but enable flexible specification of the participation requirements. Third is the need to identify and reify a suite of interaction patterns representing positions on a spectrum between completely passive sensing and active sensing. Each of these patterns will bring together a subset of the standardized protocols and language specifications and describe a generalized, reusable solution to defining the crowdsource that can be tailored for a particular application.

### 2.2 Maintaining Privacy in the Crowdsource

Anytime applications ask users to contribute information to be shared and accessed by other (potentially unknown) users, privacy must and should be one of the premier concerns. While many existing applications rely on users' desire to provide information that can help social and environmental causes, such applications lack the ability to scale beyond niche extraordinarily exuberant participants. Many participatory sensing applications require logging information associated with users' locations (e.g., potholes or traffic the user's vehicle encounters); releasing this information has the potential to release private information about the individual that he or she may not want publicly available.

A framework to support the design and development of participatory sensing applications must consider these privacy concerns and must provide programming tool support to control privacy and evaluate applications for privacy preservation [18]. As in other domains, we specifically promote the development of *usable* privacy controls tailored to the needs of participatory sensing applications. Within the participant recruitment and maintenance policy, participants must be given easy-to-use control over their information and how and with whom it is shared. In addition, approaches such as aggregating data from multiple participants can increase the privacy of individuals [5], and such approaches must also be a part of participatory sensing software development frameworks.

## 2.3 Evaluating and Improving the Crowdsource

Participatory sensing applications rely on the crowdsource to enable decision-making. It is essential, then, that applications are provided with constructs that expose the *fidelity* of data collected from participants and that support decision-making processes. The fidelity, or quality, can be defined among several dimensions:

- *Fidelity of the sensor.* Sensors, even purposed ones, are not perfect. They may be miscalibrated or malfunctioning, providing erroneous data. In addition, there is a spectrum of the quality of reading that is obtainable from sensors of the same type. These imperfections are magnified in participatory sensing applications where participants' different devices may have different capabilities. For example, the capabilities of cameras on various mobile phones often differ in terms of the frame rate, image resolution, and pan/tilt/zoom options. For some applications, the humans themselves are the sensors (e.g., in applications that ask participants to log observations [2]), introducing even more variability into the quality of the sensed data.

- *Fidelity of the provider.* Even when the humans are not the sensors, the human participants are often in direct control of the sensor. Human participants are not perfect, either. They may use the sensors improperly to sample data requested by a participatory sensing application. For example, in a participatory sensing application that collects citizens' videos of traffic to estimate pollution in an area, a participant may record video for a shorter duration than requested. Participants may also provide invalid or stale information, perhaps even with malicious intent.

- *Fidelity of the collection process.* As data is collected, participants may move or even disconnect from the network. When a participant moves during a distributed data collection process, it may result in the data being collected multiple times or not at all. While this has been investigated from a pervasive computing perspective targeted to purposed sensing deployments [12], participatory sensing applications are likely to encounter extreme dynamics with respect to mobility and disconnection of participants.

Measuring the fidelity of the data in a participatory sensing application can be as difficult as defining the meaning of fidelity in the first place. As in any application, the quality of the data collected depends on the potential use of that data after collection. It is often possible to trade off such fidelity for decreased cost in terms of number of required participants, quality of required sensors, etc. Before we can explore these trade offs, however, we must first have a concrete, formal grasp of the meaning of fidelity in these participatory sensing applications

Specifically, we need metrics that can be used to assess the fidelity of the crowdsource along the three dimensions described above. Recent work on data fidelity [1, 7, 19], provider fidelity [13], and collection fidelity [12] has introduced basic principles forming a framework for fidelity discovery in distributed and dynamic environments. However, given the new paradigm of participatory sensing, we must extend these principles and develop a suite of fidelity metrics that are sensitive to the particular constraints of participatory sensing applications. As just one example, we may want to attempt to define a *confidence* metric for an aggregate data value fused from a set of crowdsourced data. Work on metrics for the fidelity of participatory sensing applications will define a concrete set of such metrics. Applying the resulting metrics would provide a measure of the fidelity of the individual elements of the crowdsource and would indicate how the fidelity of the part impacts the fidelity of the crowdsource as a whole.

It may be the case that the crowdsource is found to be inadequate in terms of the achieved fidelity. As a result, the participatory sensing application may need to provide feedback to participants to improve sensed data. The definition of the crowdsource may also need to be refined. We must also consider the development of tools that help software engineers to evaluate these metrics and to make decisions about the crowdsource. Visualization tools may prove to be especially useful in exploring the reasons behind poor values of fidelity metrics and determining how to adjust the definition of the crowdsource in response. Finally, we must go beyond the use of metrics for post hoc analysis and develop new design patterns that document how to perform real-time, in-network fidelity assessment that is integrated within a participatory sensing framework. With this kind of evaluation, application developers can design participatory sensing applications that can dynamically adapt and refine the definition of the crowdsource.

## 2.4 Integrating the Crowdsource with Purposed Sensing

As interest in sensor networks has boomed over the past decade, a number of applications have been deployed in the field. As noted in [20], most deployments are targeted to meet the needs of a single application, providing sensing or actuation capabilities that are specific to its particular tasks. We refer to these kind of deployments as *purposed sensor networks.* If the vision of the sensor network research community comes to fruition (which is increasingly the case), purposed sensors will be ever more ubiquitous.

Applications that rely on participatory sensing should be designed to take advantage of purposed sensing when it is available and applicable. Often, participatory sensing applications share similar goals with those deployed in purposed sensor networks. For example, a participatory sensing application may monitor pollution in an urban area by using videos submitted by mobile phone users to count the number of vehicles that were seen within a given area and timeframe [3], while at the same time, a sensor network application may use purposed sensors to detect levels of harmful emissions in the city. Similarly, purposed sensors on bridges can monitor their condition with respect to upkeep and vibration properties, while participatory sensing applications can get a feel for what real users are experiencing in their automobiles [4, 10]. We posit that combining data from a crowdsource and from purposed sensors can lead to better results. Even when the application's connection to a purposed sensor network is fleeting, sensing coverage is sparse, or the purposed sensors match only a subset of the application's needs, the data collected from purposed sensor networks can be used to validate the crowdsource data obtained through participatory sensing and vice versa.

To support the integration of these sensing modalities, we must address challenges in discovering and reconciling deployed purposed sensor networks that match the application's needs. We must first revisit specification languages and libraries that enable definition and maintenance of the crowdsource. Since purposed sensors use a different type of sensor than a human participant to capture the same or similar data, specification languages must be flexible enough to specify the sensor while making the distinction between human and purposed sensors transparent to the programmer. Supporting libraries must be able to reconcile differences in granularity, locality, and fidelity of sensed data between participatory and purposed sensors. Approaches to multimodal data fusion must be incorporated within a participatory sensing framework, and live debugging tools are needed that provide visualizations of the origination and in-network fusion of information. Finally, we must consider how to incorporate existing frameworks for data acquisition in sensor networks with elements of a participatory sensing framework. To do so requires a careful study of potential architectural mismatches across these paradigms, as well as the need for new architectural connectors.

# 3. CONCLUSIONS

Participatory sensing is poised for explosive growth. It is supported by commodity devices that are already in widespread use, and the idea of contributing their own perspectives, opinions, and data appeals to the altruistic nature (as well as the ego) of participants. It is essential that we invest in the development of software engineering principles, techniques, and tools to support the integration of participatory sensing in application development. In this paper, we put forth research challenges and directions that the software engineering community will face as it examines the formation of an infrastructure to support participatory sensing applications. This provides a starting point for identifying a principled approach to engineering participatory sensing software and moving the practice into the mainstream.

# 4. REFERENCES


[1] M. Bawa, A. Gionis, H. Garcia-Molina, and R. Motwani. The price of validity in dynamic networks. In *Proc. of ACM SIGMOD*, pages 515–526, June 2004.

[2] Project BudBurst. `http://budburst.ucar.edu`.

[3] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Shrivastava. Participatory sensing. In *Proc. of the Workshop on World-Sensor-Web: Mobile Device Cecntric Sensor Networks and Applications*, pages 117–134, 2006.

[4] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan. The pothole patrol: Using a mobile sensor network for road surface monitoring. In *Proc. of the $6^{th}$ Int'l. Conf. on Mobile Systems, Applications, and Services*, pages 29–39, 2008.

[5] R. K. Ganti, N. Pham, Y.-E. Tsai, and T. F. Abdelzaher. Poolview: Stream privacy for grassroots participatory sensing. In *Proc. of the $6^{th}$ ACM Conf. on Embedded Network Sensor Systems*, pages 281–294, 2008.

[6] S. Gaonkar, J. Li, R. R. Choudhury, L. Cox, and A. Schmidt. Micro-blog: Sharing and querying content through mobile phones and social participation. In *Proc. of the $6^{th}$ Int'l. Conf. on Mobile Systems, Applications, and Services*, pages 174–186, 2008.

[7] N. Jain, D. Kit, D. Mahajan, P. Yalagandula, M. Dahlin, and Y. Zhang. Network imprecision: A new consistency metric for scalable monitoring. In *Proc. of OSDI*, pages 87–102, 2008.

[8] N. D. Lane, S. B. Eisenman, M. Musolesi, E. Miluzzo, and A. T. Campbell. Urban sensing systems: Opportunistic or participatory? In *Proc. of the $9^{th}$ Workshop on Mobile Computing Systems and Applications*, pages 11–16, 2008.

[9] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. M, 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 the $6^{th}$ ACM Conf. on Embedded Network Sensor Systems*, pages 337–350, 2008.

[10] P. Mohan, V. N. Padmanabhan, and R. Ramjee. Nericell: Rich monitoring of road and traffic conditions using mobile smartphones. In *Proc. of the $6^{th}$ ACM Conf. on Embedded Network Sensor Systems*, pages 323–336, 2008.

[11] 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 the $7^{th}$ Int'l. Conf. on Mobile Systems, Applications, and Services*, pages 55–68, 2009.

[12] J. Payton, C. Julien, G.-C. Roman, and V. Rajamani. Semantic self-assessment of query results in dynamic environments. *ACM Transactions on Software Engineering and Methodology*, 19(4), April 2010.

[13] S. Reddy, J. Burke, D. Estrin, M. Hansen, and M. Srivastava. A framework for data quality and feedback in participatory sensing. In *Proc. of the $5^{th}$ Int'l. Conf. on Embedded Networked Sensor Systems*, pages 417–418, 2007.

[14] S. Reddy, D. Estrin, and M. Srivastava. Recruitment framework for participatory sensing data collections. In *Proc. of the Int'l. Conf. on Pervasive Computing*, pages 138–155, 2010.

[15] S. Reddy, K. Shilton, J. Burke, D. Estrin, M. Hansen, and M. Srivastava. Using context annotated mobility profiles to recruit data collectors in participatory sensing. In *Proc. of the $4^{th}$ Int'l. Symposium on Location and Context Awareness*, pages 52–69, 2009.

[16] S. Reddy, K. Shilton, G. Denisov, C. Cenizal, D. Estrin, and M. Srivastava. Biketastic: Sensing and mapping for better biking. In *Proc. of the $28^{th}$ Int'l. Conf. on Human Factors in Computing*, pages 1817–1820, 2010.

[17] Sense the Beach. `http://www.sensethebeach.com`.

[18] K. Shilton. Four billion little brothers?: Privacy mobile phones, and ubiquitous data collection.

*Communications of the ACM*, 52(11):48–53, November 2009.

[19] A. Singla, U. Ramachandran, and J. Hodgins. Temporal notions of synchronization and consistency in Beehive. In *Proc. of SPAA*, pages 211–220, 1997.

[20] J. Stankovic. When sensor and actuator networks cover the world. *ETRI*, 30(5):627–633, Oct. 2008.

[21] What's Invasive! `http://whatsinvasive.com`.

[22] What's Noisy! `http://whatsnoisy.appspot.com`.