Tapping into the Vibe of the City Using VibN, a Continuous Sensing Application for Smartphones

Emiliano Miluzzo[†], Michela Papandrea[§], Nicholas D. Lane[‡], Andy M. Sarroff[†], Silvia Giordano[§], Andrew T. Campbell[†]

[†]Computer Science, Dartmouth College, Hanover, NH, USA

[§]SUPSI, Manno, Switzerland [‡]Microsoft Research Asia, Beijing, China

ABSTRACT

We present VibN, a mobile sensing application deployed at large scale through the Apple App Store and the Android Market. VibN has been built to determine "*what's going on*" around the user in real-time by exploiting multiple sensor feeds. The application allows its users to explore live points of interest of the city by presenting real-time hotspots from sensor data. Each hotspot is characterized by a demographics breakdown of inhabitants and a list of short audio clips. The audio clips augment traditional microblogging methods by allowing users to automatically and manually provide rich audio data about their locations. VibN also allows one to browse historical points of interest and view how locations in a city evolve over time. Additionally, VibN automatically determines a user's personal points of interest, which are a means for building a user's breadcrumb diary of locations where they have spent significant amount of time. In this paper, we present the design, evaluation, and results from the large scale deployment of VibN through the popular Apple App Store and Android Market.

Author Keywords

Applications, Smartphone Sensing.

ACM Classification Keywords

C.2.4 Distributed Systems: Distributed Applications.

General Terms

Design, Experimentation, Performance.

INTRODUCTION

Sensor-enabled smartphones are becoming a mainstream platform for researchers to collect information-rich data because smartphones allow the characterization of human activity and context at scale [11, 8, 2, 12]. We believe that continued research in smartphone sensing will allow us to characterize people, places, and communities as never before possible. As a case example, CenceMe [11] is an application that infers a person's activity and context using multiple sensors in a mobile phone. Figure 1 shows CenceMe data collected in Hanover, New Hampshire, over a month across 20 users. Aggregate activities such as sitting, standing, walking, and running are represented by colored markers. We can easily examine the geographic distribution of basic human activities and reason about location relationships. For instance, the red circle in Figure 1 marks the Computer Science department at Dartmouth College, which is mainly characterized by the "sitting" inferred state. CenceMe's inference is in accordance with the nature of a computer science department, i.e., the department is in an office building where people are mostly sitting during their work hours. The CenceMe example helps us understand the significance of collecting data using a continuous sensing application running on multiple smartphones. We are given the opportunity to characterize spaces at a very fine grained level, which is generally impossible without burdensome subject polling. Such information may be useful, for example, to help city managers understand how people exploit urban spaces, resulting in improved urban planning. Alternatively, physicians may learn the health behavior of a community and use this information for community health assessment and recommendations. Distributed sensor monitoring and inference can provide real-time insights, augmenting inter-person interaction, as well as interactions between people and spaces. Questions we may answer include: what music is being played at a particular club right now, how many people are at the club, and what are their demographics? Where is the quietest place in the city to read a book? How many people are jogging in the park right now, so that I won't be alone during my run today?

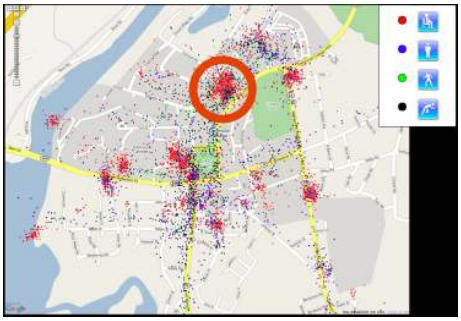


Figure 1. CenceMe inference data generated over a month across 20 subjects in Hanover, New Hampshire.

SCI'11, September 18, 2011, Beijing, China.

This paper presents VibN [1], a continuous sensing application for smartphones. The goal of VibN is to answer questions such as those posed above by collecting sensor data, executing inferences, and presenting results to users that inform them in real-time about "what's going on" around them. VibN automatically provides structured visual information about the places people spend their time by displaying real-time hotspots of the city, which we call Live Points Of Interest (LPOI). We think this paradigm poses vast improvement over other models that are constrained by manual input, such as in the Wertago application. A LPOI, which is derived from a backend clustering algorithm, is represented by the demographics of its inhabitants, such as average age, ratio of men and women, and their relationship status. VibN allows its users to replay historical LPOIs, encouraging observation on how hotspots and their demographics evolve over time. In this work, we define a point of interest as any location that people spend a significant quantity of their time. Thus, places of work, living, and entertainment are points of interest. VibN also introduces a new dimension to microblogging through the *Vibe it!* feature, which allows a user to record audio commentaries. Audio input provides richer means of building understanding about locations, activities, and events than the current short text microblogging model. However, because VibN is founded on an opportunistic sensing paradigm [4]—where the user is not an active participant in the sensing phase—it also transparently records short audio clips in the background to provide near continuous audio context. Segments of audio containing voice are filtered out from the clip to preserve the user's privacy. VibN also automatically and transparently builds a personal diary, giving the user an opportunity to track locations of significance and related audio vibes. This paper will discuss our system design, implementation, and evaluation of VibN.

In summary, these are the key contributions of our work: 1) We show that VibN, by running continuously in the background on smartphones, is able to characterize the way people and communities interact with the locations they inhabit. 2) We present the design, implementation and evaluation of VibN, which has been deployed at large scale through the Apple App Store and the Android Market, and used by over 1000 users in six months of operation.

DESIGN

In this section, we present the design of the phone and backend architecture of the VibN application.

Phone Client

The client design is modular and can be implemented on both the iOS and Android platforms according to the same principles of flexibility and efficiency. In the following, we describe VibN's components and design principles, followed by native differences that impact the implementation.

Sensing. We use accelerometer, audio, and localization sensor data for our application. A sensing manager activates sensors according to the directives of a duty cycling manager. All data is sensed transparently, except for audio sensing, which can also be activated by the user. Transparent sensing occurs in the background without the user's active participation. Background audio "vibes" are introduced to periodically capture a person's context. The *Vibe it!* feature allows active participation by the user, in which they can initiate recording of a short audio clip. Every *Vibe it!* clip is geo-tagged before being uploaded to the server backend. All sensor data is handled by two components: the personal data manager, which is responsible for the personal diary points of interest; and the communications manager, which handles bi-directional communications with the VibN server. The iOS and Android platforms have different methods for handling their native location and audio engines. VibN has been adapted to accommodate these differences.

(a) Personal view on the iPhone.

(b) Personal details on the iPhone.

Figure 2. VibN personal view and personal details view on the iPhone.

Duty Cycling Manager. The Duty Cycling Manager orchestrates sensing-sleeping cycles in order to optimize resource usage. It is important to carefully allocate duty-cycles for mobile sensing applications in order to conserve resources, in particular battery power. We emphasize localization engine (GPS, WiFi, and cellular) regulation, the continuous use of which can dissipate a phone's battery within a few hours. VibN is not designed for continuous location tracking; its goal is to identify significant points of interest. We conjecture that people tend to spend at least 30 minutes at a time at important locations, such as the home, work, gym, restaurants, clubs, etc. We leverage this assumption and design the duty-cycling algorithm to activate the localization engine after long intervals (between 30 minutes and 1 hour) and report data to the server only if the location has remained static. In this way, we maximize the likelihood that the system captures locations that are visited for intervals longer than the sensor's sleep cycle, while ignoring places visited for short intervals. There are two advantages to our approach: it extends battery lifetime by applying long sleep cycles; and it promotes data pre-filtering for the server-side LPOI clustering algorithm.

Personal Data Manager. This module manages the user's personal diary by: determining if a data point (location, or

(a) Live view on the iPhone.

(b) Historical view on the iPhone.

Figure 3. VibN live and historical views on the iPhone.

location plus audio vibe clip) is a new significant location for the user; and inserting the new data point into the personal local data cache. The personal data cache is built according to a first-in first-out (FIFO) queueing policy.

The personal data manager determines the significance of a location by analyzing the duration of a user's visit. If the visit exceeds a time threshold, then the manager flags the location as significant. We empirically use a fixed threshold of two hours, which we believe to be reasonable considering that people often visit significant locations such as the office or home for longer. We realize that this policy may not generalize well to all users, since people have different living habits and styles. Future releases will give users direct control over this parameter.

The VibN personal view for iOS is shown in Figure 2. The personal view allows a user to examine their life pattern. Green blobs (Figure 2(a)) indicate locations that the system deems significant. By tapping a green blob one can examine personal activity details, for example view what times a location was visited and listen to audio vibes recorded there (Figure 2(b)).

LPOI Manager. The LPOI manager maintains up-to-date live and historical points of interest on the phone and partitions them by time windows (see Figure 3). Points of interest are refreshed in two cases: when the application is being launched; or when the application resumes from the background. Upon refreshing, the application automatically downloads points of interest co-located near the user.

Along with the audio vibes, a point of interest is characterized by the demographics of its visitors. Demographic metrics include average age, average relationship status, and gender ratio. In the current implementation, demographic information is manually provided by users in the application's settings. In the future, we plan to leverage sensor data to automatically infer demographic data. For instance, voice pitch detection may be used to infer gender.

As the LPOI manager receives points of interest from the server, it partitions them according to time. A "live" bin receives points of interest derived from activity in the last hour. Historical bins are used to replay LPOI evolution over time (up to a month in the current implementation) by means of a graphical slider. The historical view allows easy identification, examination, and comparison of consistent hotspots versus transiently popular locations. By looking at a point of interest's details, we may observe the current demographics of a hotspot, e.g., a LPOI could be characterized by 50% males, with a mean male age of 33, mean female age of 26, and 50% single status.

User Feedback Manager. User studies, in which users are asked to report on their experience or to suggest new features, are necessary to assess the performance of a system. However, it is not always possible to collect the same quality data for large-scale deployments as for small and medium scale projects, as we have less control over compliance and it takes more time to distribute surveys over a large population. VibN's solution is the User Feedback Manager, which can dynamically survey users by presenting questions directly to the client. We are able to push down new survey questions from the server as new needs arise. Answers are uploaded to the backend providing us immediate access to important usability data.

Differences between VibN iOS and VibN Android

While the VibN iOS and Android implementations respect the high level architectural design guidelines of the system discussed above, these platforms present differences in some of their basic low level functions. In particular, the respective platforms handle localization and accelerometer management differently. These functions are dealt with separately for each platform.

Localization. The Android location engine is more flexible than the iOS counterpart. It allows the programmer to individually enable localization components such as GPS, WiFi, and cellular. This makes it easier to optimize resource usage, in particularly power. Phone resources demand careful handling when designing continuous mobile sensing applications, and the individual management afforded by Android provides increased flexibility. The iOS, however, provides less fine grained control. The programmer must specify a desired localization accuracy, which is parametrized in three levels: low, medium, high. The iOS itself decides which localization components to use in order to meet the accuracy requirements. This lack of low-level control hinders thoughtful resource management by the programmer.

Accelerometer. Smartphone' sensors have been mainly introduced to enhance the user experience when interacting with the devices, e.g., flipping the user interface from landscape to portrait mode with the accelerometer. For this reason iOS currently shuts down the accelerometer when an application is pushed to run as background process since there is no active user interface that needs the accelerometer support. The consequence of this approach is the impossibility to rely on a continuous accelerometer data stream, which is the foundation for reliable activity inference. Android OS, instead, maintains the accelerometer active even when the application is sent to the background.

Backend

Data Collection. The VibN phone client interacts with the backend using standard web service interfaces supported by the Python *web.py* framework under a standard Linux distribution. The VibN data, which we designate as "vibes," consists of the following: 1) location only vibes; 2) audio vibes captured by the application automatically; and 3) audio vibes generated by the *Vibe it!* feature.

In order to preserve the privacy of users we treat automatically sampled audio vibes differently than the *Vibe it!* audio vibes. When initiating an audio recording with *Vibe it!* a user implicitly acknowledges that data collection is taking place. However, background audio vibes are generated without user participation. For this reason, we apply an algorithm that anonymizes audio vibes automatically recorded by the phone. The algorithm removes short portions of audio from the audio stream at regular intervals so that background sounds can be identified (e.g., the sound of a car or music) but the content of conversations cannot be reconstructed.

Clustering Engine. The clustering algorithm runs on the servers, asynchronously to client queries, and it is based on the density-based spatial clustering (DBSCAN) technique. The reason for the adoption of DBSCAN is that, by being density based, it operates in an unsupervised manner without requiring the number of clusters to be computed as input parameter like for K-Means. Clustering runs are processed on a location tile and a time window. We use location tiles of size 120 by 120 km and time periods ranging from 3 hours to as long as 1 month. The output of a clustering run is a set of points of interest, stored as a record in the indexing service.

Scaling. To handle scale and guarantee backend robustness we use Amazon Elastic Cloud services. The advantage of the elastic cloud service is that machines can be promptly instantiated or terminated based upon demand. This is a desirable feature when the user base can change over time, and rapid adjustments to the backend might be needed to accommodate the application's demand.

SECURITY, PRIVACY, AND TRUST

Security, privacy, and trust are important matters for mobile sensing applications. As such, VibN attempts to ensure a secure and trustworthy system with the following steps. Personal diary data never leaves the phone and the user has full control over it. Uploaded data is stripped of any details that could reveal a person's identity. Details on live points of interest are an aggregate representation of a location without exposing any individual's information. Data sent and received over the wireless link is protected by SSL encryption. Users can disable the sensors at any time. Background audio recordings are automatically stripped of vocal content in order to preserve conversational confidentiality.

EVALUATION

VibN is implemented on iOS and Android and is able to run on Apple iPhone/iPod Touch devices, as well as multiple Android phones. The application was released to the public through the Apple App Store and the Android Market on November 18, 2010. In approximately 6 months, 1000+ users have downloaded and used the application continuously. In this section, we present a system characterization of VibN, and a characterization derived from the large data set collected from the app store users. As far as we know, this is the first characterization of a mobile sensing application released at large scale through the app stores.

System Performance

Since the location engine on these devices is the main battery drain, we focus on the battery life as a function of localization duty-cycles. From our experiments, we derive the optimal location engine duty-cycle time to be 30 minutes. After several weeks of application use, we determined this to be the interval that minimizes battery usage while collecting significant points of interest. With a 30 minute sleep cycle, the iPhone 4 battery duration is on average about 25 hours, versus 40 hours on the Nexus One. The reason that the Nexus One has a longer battery life is that Android provides native APIs to actively regulate the localization components. This gives the developer flexibility to build more power-efficient applications.

Personal Points of Interest

Personal points of interest are generated in two different ways: when the application runs in the background, and when a user records a *Vibe it!* audio clip. In both cases, given the localization error (which is larger indoors), we have to ensure that the system does not create different points of interest for the same physical location. In order to achieve this goal, a dampening scheme is required. It accepts new points of interest only if they lie outside a bounding box centered on the person's location. The bounding box must be dimensioned properly so that significant places are generated when the person moves to nearby locations and a new significant place is warranted. We evaluate the accuracy of points of interest placement in indoor locations for different dampening box sizes, ranging in radius from 11 to 60 meters. When the dampening region has a diameter of about 60 meters and the user moves from location 2 to location 1 in an adjacent building, the significant point of interest for Location 1 is not captured by VibN because it is within the dampening region. We therefore set the dampening region radius to be 11 m so that the two locations can be distinguished. We find this value to be robust across multiple indoor locations in different locations. We are planning to introduce an adaptive dampening policy based on the localization error in the future.

Backend Clustering

In this section we discuss the performance of the DBSCAN clustering algorithm running in the backend to compute LPOIs. DBSCAN takes two parameters: the scope of the clustering (eps) and the minimum number of data points (k) necessary to form a cluster. The algorithm's performance as a function of several parameter values is shown in Figure 4. The raw vibe locations, uploaded from seven different locations, are reported in Figure 4(a). After several experiments, we pick k=5 and eps=0.002, which allows better clustering accuracy and minimizes false positives. In fact, when a location is significant, several data points can be found for that place.

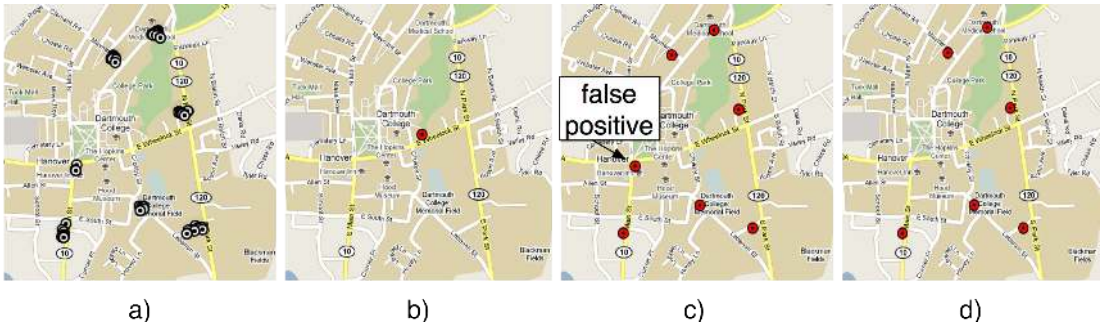


Figure 4. Backend clustering algorithm performance: a) raw location data from seven different places; b) result of the clustering algorithm with k=1 and eps=0.1; c) result of the clustering algorithm with k=1 and eps=0.02; d) result of the clustering algorithm with k=5 and eps=0.002

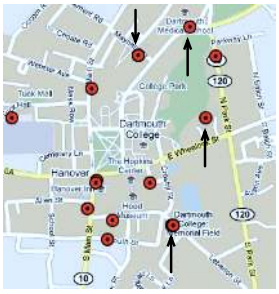


Figure 5. Spurious clusters caused by continuous location data upload.

We fix the minimum threshold to 5 data points to dampen the impact of false positives.

We also show the positive impact of the phone data pre-filtering algorithm. Continuous data uploads would generate spurious clusters, which are the result of false positives generated by sparse location data. Figure 5 shows the results of multiple phones continuously uploading data. Uploads are at intervals of less than a minute along a path that includes the locations reported in Figure 4(a), while stopping for significant times in the locations of Figure 4(a). The density of the vibes is larger in the original seven locations and lower along the path. It can be seen from Figure 5 that the sparse data along the path, which doesn't represent points of interest, create spurious clusters. Only four of the original LPOIs can be identified (indicated by arrows in Figure 5). Hence the pre-filtering approach on the phone boosts the clustering performance.

VibN Usage Characterization

We present application usage statistics by analyzing data collected from the Apple App Store and the Android Market for more than 1000 users as of the time of writing.

Demographics. The VibN users' age distribution is shown in Figures 6(a) . It is interesting to observe that the average VibN users' age is below 30.

Device breakdown. The fraction of Android versus iOS users is shown in Figure 6(b). It is interesting to see that the number of Android users is larger than the number of iOS users. We believe the reason is that Android OS is supported by many different smartphone models compared to iOS, available only for Apple smartphones. With its more flexible programming platform and absence of review process, Android becomes a very appealing platform for researchers to quickly roll out mobile sensing applications at scale.

Table 1. Fraction of users allowing their data to be used for research purposes.

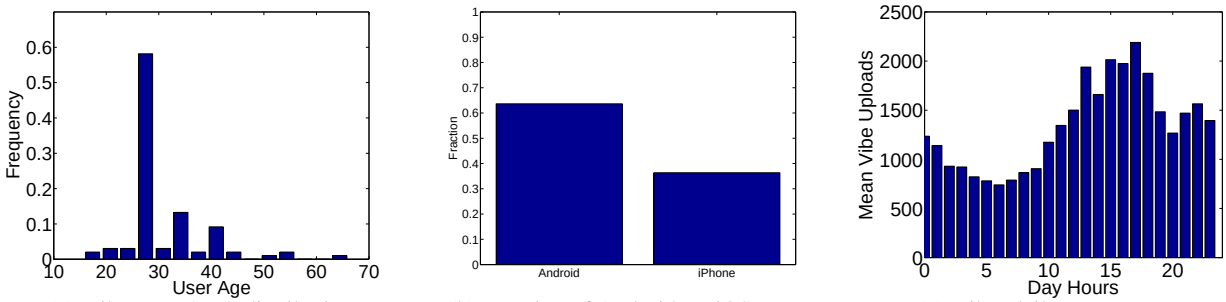
Participating	Not participating
25%	75%

Usage pattern. It is important to identify the application usage pattern in order to design a system that is flexible enough to be responsive when necessary, for example to handle bursts of users. In particular, by knowing when users are mostly active (see Figure 6(c)), we design the VibN backend in order to: instantiate more machines to accommodate high loads during the day, and make the clustering algorithm more responsive during peak hours.

Privacy Settings. In order to use data for research purposes, it is necessary to comply with the directives of the Institutional Review Board (IRB) university committee, which requires users to be informed if their data is going to be used for academic research. To this end, we add an informative text following the terms of service when the application is downloaded asking the user whether they would like to participate. The breakdown of voluntary user participation versus non participation is reported in Table 1. These numbers give an important message: it is still unusual for people to find applications designed for research purposes on commercial app store distribution systems. Consequently, by not fully understanding the mechanism and the risks involved, people simply opt-out from participating. Convincing people to participate in research remains a challenge. This may lead to slow user base growth and data collection process.

RELATED WORK

Smartphones are becoming a mainstream platform for realizing mobile sensing applications and frameworks at scale [2, 5, 9, 15, 3, 16, 19, 11, 12, 13, 10]. Several techniques



(a) VibN users' age distribution. (b) Fraction of Android Vs iOS users. (c) VibN daily usage pattern.

Figure 6. VibN usage characterization.

to optimize the usage of the phone's resources for continuous mobile sensing applications have recently been presented [18, 14, 7]. Researchers consider mobile sensors a scalable way to collect sensor data without the need of a fixed infrastructure by using ad-hoc sensors on moving objects such as bikes [6], or smartphones with an application for audio noise mapping [16]. These are examples of techniques useful to derive sensor data maps of places and cities in a scalable fashion. There is continuous growth of applications designed to promote awareness of city events, or as a means of socially connecting people based on location, e.g., Foursquare.These applications are usually user input driven. More recent applications, such as TweetMic, allow association of audio clips with individual Twitter accounts. Overall, these applications share a similar goal, which is to meet the increasing interest in gathering real-time information about places and to more efficiently take advantage of what a city has to offer. The goal of VibN is to meet the demand for real-time rich content information by exploiting continuous sensing on smartphones. Researchers [17] have already started to realize the opportunity behind using large scale application distribution systems (such as app stores) to collect data beyond the boundaries of a research lab. A study showing how to apply a multi-modality sensing approach to correct localization error has been shown in [2].

CONCLUSION

In this paper we presented the design, implementation, and evaluation of VibN, a continuous sensing application for smartphones. We discussed the implementation of VibN for the iOS and Android platforms and showed its performance on the Apple iPhone 4 and Google Nexus One. We presented the characterization of the application from the release of VibN to the public through app stores such as the Apple App Store and the Android Market. We reported on the characterization of the application from the usage of over 1000 users using it worldwide.

ACKNOWLEDGEMENTS

This work is supported in part by Nokia, Intel Corp., Microsoft Research, and NSF NCS-0631289.

REFERENCES

1. VibN Portal. http://sensorlab.cs.dartmouth.edu/vibn.
2. M. Azizyan and et al. SurroundSense: Mobile Phone Localization via Ambience Fingerprinting. In *MobiCom'09*, 2009.
3. X. Bao and R. Choudhury. MoVi: Mobile Phone based Video Highlightsvia Collaborative Sensing. In *MobiSys'10*, 2010.
4. A. Campbell, S. Eisenman, N. Lane, E. Miluzzo, and R. Peterson. People-centric urban sensing. In *WICON'06*, 2006.
5. T. Das and et al. PRISM: Platform for Remote Sensing using Mobile Smartphones. In *MobiSys'10*, 2010.
6. S. Eisenman and et al. BikeNet: A mobile sensing system for cyclist experience mapping. In *SenSys'07*, 2007.
7. D. Kim and et al. SensLoc: Sensing Everyday Places and Paths using Less Energy. In *SenSys'10*, 2010.
8. H. Lu and et al. SoundSense: Scalable Sound Sensing for People-Centric Applications on Mobile Phones. In *MobiSys'09*, 2009.
9. H. Lu and et al. The Jigsaw Continuous Sensing Engine for Mobile Phone Applications. In *SenSys'10*, 2010.
10. A. Madan and et al. Social Sensing for Epidemiological Behavior Change. In *UbiComp'10*, 2010.
11. E. Miluzzo and et al. Sensing Meets Mobile Social Networks: the Design, Implementation and Evaluation of the CenceMe Application. In *SenSys'08*, 2008.
12. E. Miluzzo and et al. Darwin Phones: the Evolution of Sensing and Inference on Mobile Phones. In *MobiSys'10*, 2010.
13. P. Mohan and et al. Nericell: Rich monitoring of road and traffic conditions using mobile smartphones. In *SenSys'08*, 2008.
14. M. Musolesi and et al. Supporting Energy-Efficient Uploading Strategies for Continuous Sensing Applications on Mobile Phones. In *Pervasive'10*, 2010.
15. K. Rachuri and et al. EmotionSense: A Mobile Phones based Adaptive Platform for Experimental Social Psychology Research. In *UbiComp'10*, 2010.
16. R. Rana and et al. Ear-Phone: An End-to-End Participatory Urban Noise Mapping System. In *IPSN'10*, 2010.
17. M. Rohs and et al. WorldCupinion: Experiences with an Android App for RealTime Opinion Sharing during World Cup Soccer Games. In *Research In The Large Wokshop'10*, 2010.
18. Y. Wang and et al. A framework of energy ef?cient mobile sensing for automatic user state recognition. In *MobiSys'09*, 2009.
19. G. Yavuz and et al. A Smartphone Based Fall Detector with Online Location Support. In *PhoneSense'10*, 2010.