A preliminary investigation of user incentives to leverage crowdsensing activities

Nicolas Haderer, Romain Rouvoy, Lionel Seinturier

A preliminary investigation of user incentives to leverage crowdsensing activities

Nicolas Haderer
University Lille 1 – LIFL
Inria Lille – Nord Europe, France
Email: nicolas.haderer@inria.fr

Romain Rouvoy
University Lille 1 – LIFL
Inria Lille – Nord Europe, France
Email: romain.rouvoy@univ-lille1.fr

Lionel Seinturier
University Lille 1 – LIFL
Inria Lille – Nord Europe, France
Institut Universitaire de France
Email: lionel.seinturier@univ-lille1.fr

***Abstract*—With the continuous emergence of smart portable devices, like TabletPC or smartphones, crowdsensing is becoming an active research topic. While exploiting the wisdom of the crowd provides huge benefits to pervasive applications, one of the key challenges remains to motivate people to contribute to flow of data by sharing some of the capabilities of their device. In this paper, we therefore report on the APISENSE platform, a participative platform that help scientists to collect realistic datasets from a population of voluntary participants. More specifically, we provide a preliminary investigation of how user incentives can be integrated in such a platform to encourage people to contribute while leaving enough flexibility to the scientist for choosing the model that fits their requirements.**

***Keywords*-Mobile sensing; Feature Model; SaaS; Multi-cloud; Component-based software engineering; Middleware**

I. Introduction

For years, the analysis of activity traces has contributed to better understand crowd behaviors and habits [1]. As an example, the *Reality Mining* activity traces collected by the MIT Media Lab[1] or the *Stanford University Mobile Activity TRAces* (SUMATRA)[2] have become a reference testbed to validate mobile algorithms in ad hoc settings [2]. Nonetheless, the diversity of the activity traces available in these repositories remains limited and therefore often constrains scientists to tune inadequate traces by mapping some of the parameters to their requirements. More recent approaches mine the data exposed by location-based social network like Gowalla or Foursquare, but the content of these activity traces remains limited to coarse-grained locations collected from users check-ins.

In this context, it has been demonstrated that the new generation of mobile phones can revolutionize collection of crowd activities traces. The survey in [3] identifies key points of its potential. Largely adopted by populations, with more than 472 millions sold in 2011 (against 297 millions in 2010) according to Gartner institute[3], smartphones have become a central piece in people's life. Not only focusing on computing or communication capabilities, modern mobile devices are now equipped with a rich suite of sensors enabling a new class of sensing applications that recognize user activities, their context, and environments. In addition, the generalization of *app stores* or *markets* provided by phone vendors allow scientists to deliver new sensing applications to large populations of users and thus leverages the enrollment of participants to a larger scale than it was possible previously.

Building a mobile sensing system involving a large communities of mobile users is not trivial, as it generally requires to implement the following phases. The *recruitment phase* consists in selecting a group in a population to execute a specific sensing task. The *deployment phase* is the ability to propagate the sensing task over selected participants and is followed by the *execution phase* where participants collect and report data from their mobile devices. And finally, the *incentive phase* where participants are rewarded, financially or not, for the performed task. In addition to that, due to the sensitive nature of data collected and the power limitations of mobile devices, a mobile sensing system must also support mechanisms to protect participants privacy and minimize the energy consumption of their devices.

For all these phases that constitute key challenges for the mobile sensing paradigm, we can find a plethora of models studied in the literature. Often, each model has only been elaborated for a specific case study. To the best of our knowledge, the state-of-the-art platforms implement one or part of these models, without providing any flexibility in the application design, but rather imposing a specific model which cannot be adapted in another context. For example, a mobile sensing experience allowing to build a network coverage map of a particular city does not need to store a participant identifier, while a sensing experiment aiming at observing the participants behavior in a city needs to store this information and therefore must provide different mechanisms to preserve participants privacy. We believe that this lack of flexibility prevents the wide adoption of crowdsensing by others research communities, which have specific requirements for their studies, without any expertise in mobile sensing technologies to develop their own systems and attract a large community of participants.

In this paper, we argue that current mobile sensing systems must evolve towards new research communities, which are not expert in mobile sensing systems. We are

[1] http://reality.media.mit.edu
[2] http://infolab.stanford.edu/pleiades/SUMATRA.html
[3] http://www.gartner.com/it/page.jsp?id=1924314

therefore particularly interested in discussing about *user incentive* that such a platform should implement in order *i)* to seduce scientists, and *ii)* encourage partipants to contribute to the sensing experiments. Scientist incentives include the modular integration of a wide variety of features that they can compose upon their need, while participant incentives deal with privacy and rewarding mechanisms that can be offered in order to catalyze the collection of datasets. This discussion is illustrated on the APISENSE platform, a crowd sensing platform we developed. Interestingly, we show how the cloud computing model and a component-based software architecture can be used to leverage such user incentives.

The remainder of this paper is organized as follows. In Section II, we briefly discuss of current mobile sensing model studied in the literature and limitations of current mobile sensing system architectures. In Section III, we report on the design choices of the APISENSE platform that illustrates our proposition before concluding (cf. Section IV).

II. State of the Art & Limitations

Mobile sensing has recently become an active research topic that aims at proposing new models and platforms to leverage the process of collecting data on the field. In this section, we report on the state-of-the-art models recently published and we discuss their key limitations. This discussion is organized along five criterias: the execution of sensing tasks, the description of sensing tasks, the recruitement of participants, the privacy issues, and the user incentives.

Sensing Task Execution In mobile sensing, the execution of a sensing task can follow two different models identified as *participatory sensing* or *opportunistic sensing*. Participatory sensing models used in systems like [4], [5] require explicit user actions to share sensor data (*e.g.* taking a picture, answering to a survey). In opportunistic sensing approaches, the sensing task is executed in background of the mobile devices without any explicit user involvement [6], [7].

Sensing Task Description In the literature, several domain-specific language has been proposed to describe sensing tasks. MyExperience [5], proposed for Windows mobile smartphones, provides a configuration language based on XML in order to control the features of the application. MyExperience collects data using a *participatory* approach—*i.e.*, by interacting with users when a specific event occurs (*e.g.*, asking to report on the quality of the conversation after a phone call ends). The definition of XML-based languages has also been explored by [4].

FUNF [8] is an Android toolkit focusing on the development of sensing applications. FUNF *in a box* is a service provided by FUNF to easily build a dedicated sensing application by filling in a web form to configure the data collect as well as the periodicity of the sensing task. While this system proposes a lightweight way to describe a mobile sensing task, potentially accessible to scientists with no background in computer science, the expressiveness to describe a sensing task is still limited to collected raw data. On the contrary, *Pogo* [6] proposes an *API*, based on JavaScript, to perform complex processing before reporting the collected datasets, but it requires a stronger background in programming languages to use the platform.

Participant Recruitment To recruit and deploy a sensing task over a population of participants, two strategies have mainly been studied by the community. The former, called *pull-based approach*, used in [9] is a proactive deployment strategy where mobile phones, with or without an implication of the participant, download directly a sensing task from a remote server. The latter, *push-based approach*, automatically propagates the sensing tasks to the mobiles devices of participants. PRISM [7], for example, is a system implementing this approach, and using a dedicated recruitment model to select the candidate participants depending of their location and their battery resources available to push a sensing task. In [10], a more complex model has been proposed to recruit participants based on reputation metrics.

Participant Privacy AnonySense is an opportunistic collection platform with a strong emphasis on privacy [9]. By combining novel techniques, such as *k-Anonymity* and *tessellation*, reported data collection time and location are blurred in order to preserve the privacy of participants and thus prevent attacks on geo-spacial data. Despite these techniques have been largely adopted in multiple studies [11], [12], the cost of privacy enforcement implies the degradation of the quality of data, thus also potentially decreasing its utility. Even if these kinds of protection mechanisms can be considered as sufficient in some cases, they quickly become irrelevant for studies requiring a fine-grained location of participants. SensorSafe [13] is another participatory platform providing fine-grained temporal and location access control mechanisms to keep the control of data collected by sensors on mobile phone.

Participant Rewarding In order to attract participants, a sensing platform has to provide appropriate levers to catalyze the collection of datasets. As cited by [14], one key challenge when cellphones are used as a research platform is to incite users to participate to a given experiment. Even if sensing applications represent a great interest for scientists, it does not offer any particular service to the participants, while consuming their resources (*e.g.*, battery, bandwidth). Recently, [15] proposed two models to characterize incentive mechanisms. A *platform-centric model*, where initiators of the sensing task set the price they want assigned to some data and a *user-centric model* where the price is defined by the participants. However, current models focus essentially on financial aspect to reward participants, which cannot be accessible for small scientist laboratories. We believe that other incentive models can be incorporated in mobile

sensing system, inspired by models like *Foursquare*[4], by incorporating serious games in their systems in order to attract and motivate participants.

Current solutions focus on one or part of the mobile sensing challenges. The state-of-the-art platforms impose a static choice to setup sensing experiments, thus limiting their reuse in other scientific context with different requirements. Indeed, none of the studied models represents an ideal solution, but rather strongly depends on the context (*e.g.* targeted population, type of collected data). Another important limitation resides in the server infrastructure of proposed platforms. Typical mobile sensing platforms adopt a *System-as-a-Service* deployment model hosted by an infrastructure that uses *multi-tenant* architecture style. This architecture style imposes all the users of the platform to share the platform resources, storage mechanisms, and financial model implemented by the provider.

We believe that adopting a multi-tenant approach highlights several leaks for building a flexible mobile sensing platform. By hosting all the collected data in the same infrastructure, a malicious attack or a bad implementation of security mechanisms in the application design can taint scientists data integrity, thus causing a potential *security* leak. Processing data over a large population of mobile users requires considerable computing resources, which can impact the *availability* of the platform for the other tenants. And finally, multi-tenancy restraints *platform customization*, by imposing infrastructure defined in the *Platform-as-a-Service* level.

III. APISENSE Software Architecture

To illustrate our proposition to cope with the current limitations described in the previous section, we introduce APISENSE, a mobile sensing platform, targeting various scientist communities to help them to build and deploy sensing experiments over large population of mobile users. The APISENSE architecture principles focus mainly on three aspects: *i) flexibility* of the platform to be easily extensible to address unforeseen requirements, *ii) customization* including fine-grained configuration level to define a sensing experiment, and *iii) scalability & security* of the server infrastructure.

A. Multi-cloud architecture

Figure 1 illustrates an architecture overview of our APISENSE platform. Unlike current mobile sensing platforms, APISENSE adopts a *multi-cloud* orientation using a star topology where the *central node* is assumed as the trusted part of the system.

In this topology, the entry point for scientists that want to define and deploy a sensing experiment over mobile users is the central node. New scientists can login in from the web

[4]https://foursquare.com

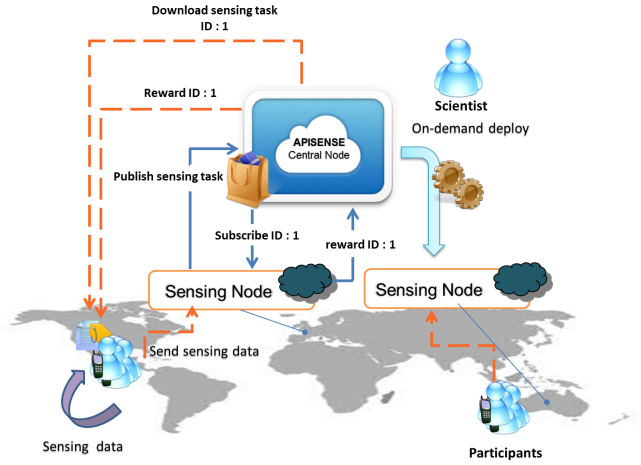

Figure 1. APISENSE Multi-Cloud Architecture

interface, and create their own SaaS instance (*i.e.*, a *sensing node*). The newly created SaaS is a dedicaded environment, providing typical services of mobile sensing system to *i)* describe experiment requirements in a domain-specific language, *ii)* deploy sensing experiments in the central node, and *iii)* connect other services to the platform to extract and reuse collected datasets (*e.g.*, visualize, analyze). From the central node, scientists can trigger the deployment (or the download) of the configured SaaS as a web archive (war) or as a virtual appliance [16].

APISENSE therefore reflects the features of the dedicated environment as a *software product line* (SPL), which is used by scientists to build their own sensing node SaaS product. This design breaks the vendor lock-in syndrome by offering the scientist the opportunity to select the IaaS or the PaaS she prefers, which can be a public or private infrastructure, depending on her preferences or on ethical and privacy issues related to the storage of collected traces. The other benefits of this design deal with *customization*, where scientists have the possibility to select an appropriate infrastructure at the IaaS level, such as supported language and specific database to store the datasets collected from mobile users. And finally, all scientists sensing node resources are completely isolated from others, improving *security* and *availability* as a scientist application can take down a server instance or perform a computation without affecting the other instances. Once deployed, the scientist can connect to her SaaS in order to create a new sensing experiment, which will be published in the central node with the experiment requirements (*e.g.* privacy and incentive mechanism, recruitment and deployment model) in order to make it available to mobile users.

Mobile phone users, identified as *participants* in the platform, can use the dedicated mobile phone application to register to the trusted central node indicating her incentive to participate to a sensing experiment. Once registered, the

mobile application sends automatically sensing capabilities of her mobile device as well as the hardware configurations. Participants can complete this information by indicating a geographical zone and a period of the day they are willing to execute a sensing task. If they activate the *push deployment model*, the central node will propagate sensing tasks in their mobile devices without their involvements. All the user profiles will remain confidential, and cannot be directly accessed by others scientist sensing nodes in the system, thus improving the protection of their privacy. The participants can also proactively retrieve a list of the available sensing experiments, depending on the participant constraints and the requirements defined by scientists. For all the experiments, information about sensing experiments configuration is provided, including privacy mechanism (*i.e.*, keep anonymous user identifier or not, privacy mechanism techniques), incentive mechanisms (*i.e.*, financial rewarding, serious gaming, goal of the experiment) and data collected on their mobile devices. Once enrolled in a experiment in the central node, an anonymous user identifier is sent to the sensing node hosting the experiment to prevent a new recruitment in the sensing experiment. This anonymous user identifier is also returned to the participant, with the SaaS URL that will be used to upload the collected data. To enforce the privacy concerns, the mobile application allows participants to adjust their privacy preferences in order to constrain the sensing task to collect data. Three categories of privacy rules can be defined. Rules related to *location* and *time* specify geographical zone or time intervals conditions under which experiments are authorized to collect data, respectively. The last category of privacy rules refer to *authorization rules*, which prevent sensors activation or access to raw sensor data if the user does want to share this information.

B. Handling variability

As defined previously (cf. Section II), a plethora of models has been proposed to realize all phases of a sensing experiment.

To cope with this diversity and propose a fine-grained configuration level to define a sensing experiment, we decided to adopt *Software Product Line* (SPL) approach [17]. SPL engineering aims at generating specific products from the requirements expressed by *customers* by composing a set of complementary features. A *Feature Model* (FM) is used to compactly define all the features in an SPL and their valid combinations. A FM is composed of a hierarchy of *features*, where a feature can be *mandatory* or *optional* and and may form Xor or Or-groups. Constraints (*e.g.*, implies or excludes) can also be specified using propositional logic to express inter-feature dependencies.

Figure 2 depicts the variability model used to customize a mobile sensing experiment. In APISENSE, FM contains mainly two configurations levels, *infrastructure level* and *sensing experiment level*. Once a scientist is connected to the web interface of the central node, she can visualize the FM and select all the features of the first level of configuration. Infrastructure variability includes essentially the configuration of the components to be deployed in the sensing node: programming languages, servlet container in charge to expose services, and database technology to store collected data. If the selected features represent a valid configuration, the central node generates a virtual appliance or a web archive (war) that the scientist can download and deploy on a cloud provider. Once deployed, the sensing node provides also a web interface where the scientist can manage all the services provided. To create a new experiment, the scientist must proceed to the second level of configuration by selecting a set features from the sensing node web interface. Once the features are selected, a new instance of sensing experiment is generated (cf. Section III-C), deploying all services for publish sensing task on the central server, store data reported by participants and process or export collected data.

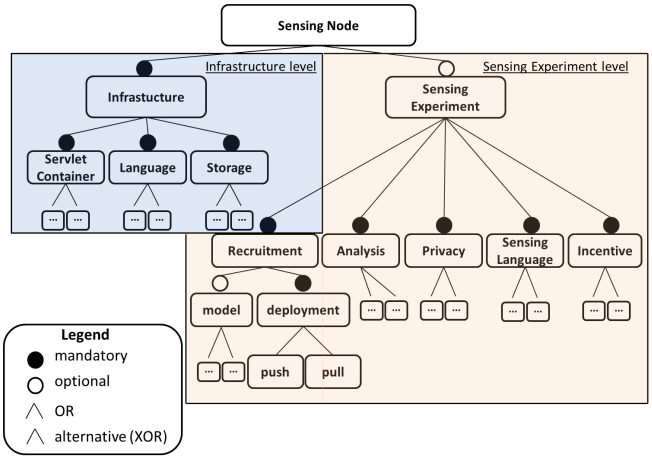

Figure 2. APISENSE Feature Model

We describe below different configuration levels proposed by APISENSE FM:

Sensing language. This variability point defines a domain specific language to describe a sensing task that will be executed by participants. We decided to adopt standard scripting languages in order to ease the description of sensing experiments by the scientists. We therefore propose the APISENSE scripting library as an extension of the JavaScript, CoffeeScript, and Python languages, which provides an efficient mean to describe an experiment without any specific knowledge of mobile device programming technologies (*e.g.*, Android SDK). Listing 1 provides a simple description excerpt, written with the languages currently supported by the APISENSE mobile application runtime, which reports the level and battery state (charging, discharging) when a new battery event is tirggered in the mobile device. The scripting

library supports a wide range of features to define data collected during a sensing experiments including traditional sensors proposed by smartphones technologies, such as GPS, Bluetooth, accelerometer, compas, phone call, sms, application status (installed, running), for *opportunistic* sensing activities and also a graphical user interaction framework to build user surveys in the case of *participatory* sensing activities. As a future work, we plan to provide also a graphical language, to allow a scientist with no background in programming languages to define her sensing tasks.

```
// Sensing language: JavaScript
battery.OnBatteryStateChanged(function(event){
  trace.add({level : event.level(),state : event.state()})
// Sensing language: CoffeeScript
battery.OnBatteryStateChanged (event) -> trace.add
  {level : event.level(),state : event.state()}
// Sensing language: Python
battery.OnBatteryStateChanged(
  lambda event: trace.add
   { level : event.level(),state : event.state()})
```

Listing 1. Sensing task examples implemented in several languages

Privacy. In addition to this script, the scientist can configure some privacy filters to limit the volume of collected data and enforce the privacy of the participants. In particular, APISENSE currently supports two types of filters. The *area filter* allows the scientist to specify a geographic area where the data requires to be collected. For example, this area can be the place where the scientist is interested in collecting a GSM signal (*e.g.*, campus area). This filter guarantees to the participants that no data is collected and sent outside of this area. The *period filter* allows the scientific to define a time period during which the experiment should be active and collect data. For example, this period can be specified as the working hours in order to automatically discard data collected during the night, while the participant is expected to be at home.

Recruitment. This phase consists in selecting the recruitment mechanism to elect a subset of participants, and to deploy the candidate sensing tasks. Two deployment models can be used. A *pull-based* approach, which is a proactive deployment strategy where participants download a sensing task from the central node. Or a *push-based* approach, which automatically propagates the sensing tasks to the mobiles devices of participants whenever it matches the sensing requirements.

Incentive To help the scientists to encourage participants to contribute to their experiments, this variability point allows them to configure rewarding mechanisms in order to catalyze the sensing of relevant datasets. As the APISENSE mobile application provides several mechanisms to control the access to sensors, the rewarding mechanism is based on the quality and the volume of datasets produced by participants. Therefore, the more sensors are activated by participants and the more datasets are uploaded, the more credits the participant receives for its involvement in the experiment. For example, the scientist can allocate more credits to the GPS sensors in order to balance the energy consumption and the privacy sensitive of this sensor. The assigned credits are then used by the scientist to provide participant rankings, involvement badges, or even coupons to reward the participants. The participant is therefore free to disable some of the sensors for privacy or energy reasons, but in this case, she will receive less credits when uploading her datasets.

Analysis This last configuration point consists in selecting features in charge of processing collected datasets. At this time, we propose analysis feature given database access, visualize geolocated data and export them in various format to be downloaded by scientists.

C. SaaS Architecture

The main objective of APISENSE is to provide to scientists a platform, which is open, configurable in order to be reused in various contexts and easily extensible to deal with unforeseen requirements. To achieve this goal, we designed a SaaS architecture of APISENSE as an SCA distributed system running on the top of FraSCAti middleware [18], a reference implementation of this open service model.

SCA promotes a vision of *Service-Oriented Computing* (SOC) where services are independent of implementation languages, remote communication technologies, interface definition languages and non-functional properties. SOC has proved to be an adequate solution for building flexible and agile software systems that are resilient to changes. SCA specifies a hierarchical component model, which means that components can be implemented either by primitive language entities or by subcomponents. In the latter case the components are called *composites*. To support service-oriented interactions via different communication protocols, SCA provides the notion of *binding*. We therefore believe that SCA provides an flexible foundation for the APISENSE infrastructure by accommodating a wide diversity of programming languages and communication protocols in order to efficiently support the variety of scientists requirements.

In the APISENSE platform, all the features previously introduced are viewed as SCA components. Once deployed on a cloud provider, a sensing node mainly contains two components (cf. Figure 3), which exposes the services that can be remotely called. The former, the Experiment Manager component, allows the scientist to generate a sensing experiment instance from a valid *Feature Model*. This component exploits the reflective capabilities of the SCA platform to generate a new *Sensing Experiment* composite which includes all the selected components in the FM. The new instance thus automatically exposes new dedicated services to enable participants to report collected data (Data collector component), publish and describe a sensing task (Recruitment component), process (Query & Export components) and configure the rewarding mechanism (Serious

game component).

The latter component, *Reconfiguration engine*, provides services for managing or adding new components in the architecture. This component represents the extension points which are used by the scientist to define custom component to process and visualise the datasets collected by participants.

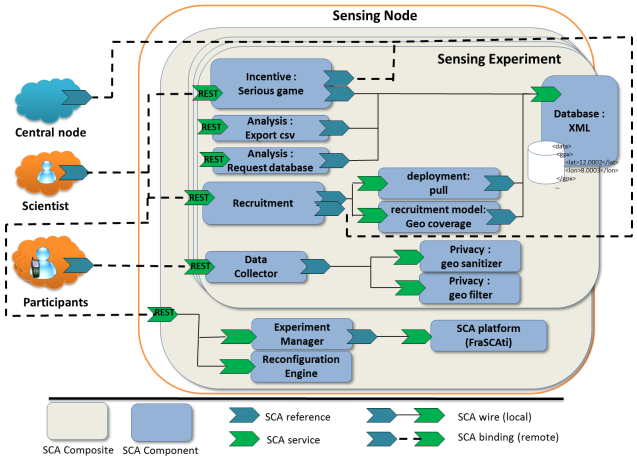

Figure 3. Sensing Node Architecture

IV. Conclusion

In this paper, we report on the design of the APISENSE crowdsensing platform. This platform distinguishes two roles: *scientists* requiring a sustainable environment to deploy sensing experiments and *participants* using their mobile devices to contribute to scientific experiments. This organization highlights two kinds of user incentives. From the scientist perspective, APISENSE is built on the principles of a multi-cloud infrastructure and offers a modular service-oriented architecture, which can be customized upon their requirements. From the participant perspective, APISENSE supports a variety of privacy mechanisms and rewarding models to choose their level of involvement in a sensing task. We therefore believe that APISENSE can provide a sustainable foundation for building a new generation of context-aware services by leveraging the collection of realistic activity traces of mobile users.

References

[1] L. Liu, C. Andris, A. Biderman, and C. Ratti, "Uncovering Taxi Driver's Mobility Intelligence through His Trace," *IEEE Pervasive Computing*, 2009.

[2] S. Ben Mokhtar and L. Capra, "From Pervasive to Social Computing: Algorithms and Deployments," in *Int. Conf. on Pervasive Services*. ACM, 2009.

[3] N. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. Campbell, "A Survey of Mobile Phone Sensing," *IEEE Communications Magazine*, vol. 48, no. 9, 2010.

[4] M. Ra, B. Liu, T. La Porta, and R. Govindan, "Medusa: A programming framework for crowd-sensing applications," in *MobiSys conference*. ACM, 2012, pp. 337–350.

[5] J. Froehlich, M. Chen, S. Consolvo, B. Harrison, and J. Landay, "Myexperience: a system for in situ tracing and capturing of user feedback on mobile phones," in *MobiSys conference*. ACM, 2007.

[6] N. Brouwers and K. Langendoen, "Pogo, a Middleware for Mobile Phone Sensing," in *13th Int. Middleware Conference*. Springer, 2012.

[7] T. Das, P. Mohan, V. Padmanabhan, R. Ramjee, and A. Sharma, "Prism: Platform for Remote Sensing Using Smartphones," in *MobiSys conference*. ACM, 2010.

[8] N. Aharony, W. Pan, C. Ip, I. Khayal, and A. Pentland, "Social fmri: Investigating and shaping social mechanisms in the real world," *Pervasive and Mobile Computing*, 2011.

[9] M. Shin, C. Cornelius, D. Peebles, A. Kapadia, D. Kotz, and N. Triandopoulos, "AnonySense: A System for Anonymous Opportunistic Sensing," *Pervasive and Mobile Computing*, 2010.

[10] S. Reddy, D. Estrin, and M. Srivastava, "Recruitment framework for participatory sensing data collections," *Pervasive Computing*, pp. 138–155, 2010.

[11] C. Bettini, S. Mascetti, X. Wang, and S. Jajodia, "Anonymity in Location-based Services: Towards a General Framework," in *Int. Conf. on Mobile Data Management*. IEEE, 2007.

[12] M. Gruteser and D. Grunwald, "Anonymous Usage of Location-based Services Through Spatial and Temporal Cloaking," in *MobiSys conference*. ACM, 2003.

[13] H. Choi, S. Chakraborty, M. Greenblatt, Z. Charbiwala, and M. Srivastava, "Sensorsafe: Managing health-related sensory information with fine-grained privacy controls," Technical Report, September 2010.(TR-UCLA-NESL-201009-01), Tech. Rep., 2010.

[14] P. Dutta, P. Aoki, N. Kumar, A. Mainwaring, C. Myers, W. Willett, and A. Woodruff, "Common Sense: Participatory Urban Sensing Using a Network of Handheld Air Quality Monitors," in *SenSys conference*. ACM, 2009.

[15] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: incentive mechanism design for mobile phone sensing," *MobiCom2012*, 2012.

[16] C. Quinton, R. Rouvoy, and L. Duchien, "Leveraging Feature Models to Configure Virtual Appliances," in *CloudCP*. ACM, 2012.

[17] K. Pohl, G. Böckle, and F. J. v. d. Linden, *Software Product Line Engineering: Foundations, Principles and Techniques*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[18] L. Seinturier, P. Merle, R. Rouvoy, D. Romero, V. Schiavoni, and J.-B. Stefani, "A Component-Based Middleware Platform for Reconfigurable Service-Oriented Architectures," *Software: Practice and Experience*, vol. 42, no. 5, pp. 559–583, May 2012.