Orchestration Support for Participatory Sensing Campaigns

Ellie D'Hondt, Jesse Zaman, Eline Philips, Elisa Gonzalez Boix, Wolfgang De Meuter
Software Languages Lab, Vrije Universiteit Brussel
Pleinlaan 2, 1050 Brussels, Belgium
eldhondt, jezaman, ephilips, egonzale, wdmeuter@vub.ac.be

ABSTRACT
In this paper we argue the need for orchestration support for participatory campaigns to achieve campaign quality, and automatisation of said support to achieve scalability, both issues contributing to stakeholder usability. This goes further than providing support for defining campaigns, an issue tackled in prior work. We provide a formal definition for a campaign by extracting commonalities from the state of the art and expertise in organising noise mapping campaigns. Next, we formalise how to ensure campaigns end successfully, and translate this formal notion into an operational recipe for dynamic orchestration. We then present a framework for automatising campaign definition, monitoring and orchestration which relies on workflow technology. The framework is validated by re-enacting several campaigns previously run through manual orchestration and quantifying the increased efficiency.

Author Keywords
Participatory sensing; measurement campaigns; abstraction; orchestration; ICT support; workflows.

ACM Classification Keywords
H.3.5. Online information services: Data sharing; Web-based services; H.5.3. Group and Organization Interfaces: Computer-supported cooperative work; Web-based interaction

INTRODUCTION

Participatory sensing [11, 7] allows people-centric environmental monitoring by way of smart mobile devices. This is by now a well-established research field as witnessed by the publication of recent surveys [4, 24]. Currently, there exist several platforms which allow people to gather data participatively. While those platforms provide a firm *technological* foundation for the domain of participatory sensing (PS), the equally important *data* and *people* aspects are much less studied. PS requires not only stable technology but also insights in methodologies for qualitative data collection and analysis, and, last but not least, engaging and keeping engaged those people that gather data using said methodologies and technology. We envision that the next-generation PS frameworks will incorporate methodologies for data gathering in a way that appeals to end-users, who are typically non ICT-experts. At the same time they should be scalable with respect to the amount of data, the number of users, and, most importantly, the type of data gathered (i.e. application area) [24, 25]. Through these frameworks, stakeholders should be able to express their concerns and see them translated into procedures for successful participatory campaigning with little or no help from platform owners. Only in this way can we move away from small-scale research-oriented deployments to the full-fledged adoption of PS as a societally and scientifically relevant method.

Recent years certainly have seen several important research outcomes in the area of methodologies for collaborative data gathering. The main question to be answered is: can the potentially enormous quantity of participatory sensing data compensate the typically inferior quality of miniature sensors used by non-experts? Several research efforts have answered that *quality-quantity* question in the affirmative [25, 6, 10]. But in order to do this, one first and foremost needs to ensure that the potential high data quantity is achieved, typically in the context of a sensing *campaign* focused in area and/or time. Additionally, organising campaigns is also essential to the people-aspect of PS. Indeed, there is high demand by communities of all sorts — grassroots, institution-led, or research-based — to be able to translate local concerns into actual campaigns for tackling them. However, the knowledge on how to organise campaigns properly is absent from currently existing PS platforms. This is problematic for stakeholders, who are largely dependent on platform owners for insider knowledge on campaign management, but also for platform owners, who are not able to scale up their efforts in sharing this knowledge.

In this article we propose a scalable system for community-friendly campaigning which encapsulates methodologies for qualitative collaborative data gathering. The design of the system is based on an abstraction of elementary campaign concepts distilled from extensive prior experience with participatory campaigning and the existing state of the art. The advantage of this approach is to act as a bridge between user-friendliness and platform reconfigurability. Indeed, on the one hand these abstractions form the building blocks for an end-user-oriented campaigning system. On the other hand, they are general enough so that we can formulate generic PS settings and thus work towards a platform that is usable in various sensing contexts. We propose a concrete implementation of this design in the area of participatory sensing of environ-

UbiComp'14, September 13-17, 2014, Seattle, WA, USA.

http://dx.doi.org/10.1145/2632048.2632105

mental noise (partly described in an earlier note [27]). Our implementation consists of components for campaign definition, monitoring, and analysis, complemented with a campaign orchestration system tasked with guiding a campaign to a successful end (in terms of data quality). To embody campaign processes in our system in a natural way, we employed workflow technology and meta-logic reasoning techniques.

Our approach specifically targets collaborative sensor efforts, though these need not be actively so. That is to say, while actively collaborative community-based campaigning is certainly one use case, campaigns may just as well be executed by loosely collected volunteer communities willing to share their data to work towards a common goal. Indirectly, we also contribute to several aspects which are argued to be highly influential for the success of next-generation systems, namely incentives for participation, dealing with faulty data and concise data representation [25], and this by way of the orchestration component which is malleable to any of the aforementioned issues.

We validate our approach by analysing 19 different campaigns which were previously deployed without ICT-support for campaign management, re-enacting 6 of them and discussing aspects of correctness, efficiency and scalability. In particular, we quantify how our system and its orchestration procedures affect management and measurement efforts of each campaign.

RELATED WORK

The term participatory sensing was first introduced by the CENS group in [11] where they present a broader vision for the application of mobile sensing technology in the context of environmental monitoring by citizen scientists. Since then a variety of participatory sensing apps has been developed, such as CenceMe [17] (social networking), BikeNet [5] (well-being) and NoiseTube [15] (environmental noise). While several review articles have shown the applicability of participatory sensing in a wide number of areas [13, 4, 2], the software technology involved is developed on a per-application basis time and again.

More recently, several articles advocate the necessity of a reconfigurable and reusable framework for mobile sensing applications. Projects such as Epicollect [3], ohmage [22], and Sensr [12] propose flexible frameworks for constructing mobile data-gathering tools. However, these frameworks limit the type of data that can be collected (typically form-based), or focus only on the campaign definition phase. Moreover, these so-called reconfigurable PS systems do not ensure that the potential high data quantity is achieved, as participants are expected to contribute freely and without guidance.

The idea of organising participatory sensing actions into *campaigns* is an obvious one and indeed has been mentioned as early as 2006 [11]. On the one hand, sensing campaigns are an answer to community concerns which are typically focused in time, place and/or other contextual factors. On the other hand, sensing campaigns are a means to ensure data density in absence of a large enough crowd contributing data. This, in turn, is essential for adequate assessment of the concern at hand. Some PS projects have focused on providing a limited form of campaign management. Examples include the use of location-based triggers [8], model-based question targeting [14], and encouraging participants to move towards desired locations [23]. All these approaches, however, lack the reconfigurable aspects of the aforementioned frameworks.

In conclusion, to the best of our knowledge, there exists no PS system which is reconfigurable and embodies notions of data quality, while covering all steps of a campaign from the design to the analysis of the outcomes. Such a system would contribute to campaign scalability as well as usability for end-users.

CAMPAIGN PROTOCOL

Despite the existence of several platforms for participatory sensing and their application in specific campaigns there is currently no formal notion of a campaign nor an operational implementation thereof. We proceed in this and the next section with the former issue, afterwards moving on to a concrete implementation of a framework to support the design, dynamic orchestration and monitoring of campaigns. The notion of data quality, crucial in the translation of campaigns from a static to a dynamic concept, is discussed in the next section.

Let us start with formalising participatory sensing (PS) platforms and campaigns running on them. PS is an inherently *active* data gathering method; measurements are gathered consciously by platform users with particular devices (typically smartphones). We denote this data collection *context* by Γ. Data is collected in separate *tracks*, where a track is a collection of measurements $\{M\}$. Finally, an individual measurement is a tuple of data points $M = (s_1, \ldots, s_n)$ where s_i are sensor readings. We take a liberal meaning of the word *sensor* here, encompassing real sensor data, user inputs, data ingested from existing datasets, or even higher-order data such as that derived by activity recognition algorithms. We write $\Gamma \Rightarrow \{M\}$ to say that the context Γ contributed dataset $\{M\}$. We then have the following definition.

DEFINITION 1. *A* campaign protocol *is a tuple of concatenated predicates $P_\Gamma + P_M$ over context Γ and measurements M respectively.*

The above predicates are themselves defined over tuples, e.g. $P_M = (P_{s_i})_i$ (which is why we use concatenation in this definition). Individual predicates are considered to be classical, i.e. they are subsets of the set of all possible outcomes for the corresponding notion. Note that P_M generally includes a temporal predicate P_T, i.e. the timeframe we are interested in, alongside a geographical predicate, which we write as P_A, with A for the area we are interested in. Here we use area to abstract away from longitude, latitude and higher-order GIS-concepts such as polygons.

As a leading example we take the NoiseTube platform [19], which enables participatory monitoring and mapping of ambient sound levels through mobile phones. In NoiseTube measurements are tuples of sensor readings for time, sound level, latitude, longitude and zero or more tags respectively, while a track is a set of such

measurements. As an example, consider the individual measurement (2011-10-13T19:32:12, 68, 50.821495,4.396498, ambulance), part of the contribution brussense1, iPhone4s $\Rightarrow$ $\{$ time$_i$, L$_{eq,i}$,latitude$_i$, longitude$_i$, tags$_i^*\}_{i=0}^n$. Here $*$ indicates that there may be zero or more tags added by the user. A city administrator who wishes to investigate sound levels recorded by his employees, using calibrated handsets, in the centre of Brussels between 8-9am during the month of May, would see his campaign formally represented as (user $\in$ {employees}, device $\in$ {calibrated-devices}, time(timestamp) $\in$ [2014-05-xT08:00:00, 2014-05-xT09:00:00] with x $\in \{1,\ldots,31\}, \top$, (latitude, longitude) $\in$ polygon(Brussels), $\top^*\}$. Here $\top$ indicates that no constraints are defined for the pertaining sensor stream, and the polygon is that corresponding to the City of Brussels.

As an aside, we obviously do not expect end-users to express campaigns in the above way, but imagine a user-friendly GUI to do this. This GUI can be built right on top of the domain-specific abstractions that we have described above. On the other hand, the above abstractions hold more generally for sensing platforms beyond NoiseTube and even beyond PS, allowing us to construct a platform reconfigurable for various sensing settings. For example, [4, Table1] lists a variety of platforms in terms of sensor modalities, all of which can be expressed as a measurement M in the above sense. The same holds for platforms such as EpiCollect [3], which, while at first sight very different due to their focus on survey-based campaigning, are actually formally identical.

CAMPAIGN QUALITY

A crucial aspect of campaigning is the notion of *quality*. Indeed, one typically translates the enormous quantity of data that is typical of PS into a more qualitative condensed representation, e.g. by location-based statistical averaging. Formalising this notion is important so that one can guide end-users to fine-tune a campaign protocol at design time. On the other hand, it allows to monitor campaign progress dynamically, triggering adapted orchestration procedures so as to ensure successful campaign completion.

Campaign quality is inherently related to *measurement density*. As an example, consider a measurement campaign where the desired outcome is a map for a particular timeframe. That is to say, we only restrict measurements according to P_A and P_T, e.g. the centre of Brussels between 8-9am as above. We then require the measurement density per unit time and area to be high enough to allow statistical reasoning. For example, in earlier work [6] we found 50 measurements per unit hour and area of $20m \times 20m$ to be an adequate sample set size. Of course, there is no exact rule for choosing the size of sample sets; rather, it is a tradeoff between measurement resources and statistical power. In PS this tradeoff is determined by the need to eliminate random errors and to increase representativeness of the dataset, while at the same time taking into account that sample sets are constructed by real people with limited time and mobility.

A concept embedded in this reasoning is that of a predicate P_{s_i}'s *resolution*, by which we mean the number of sample sets it covers. In the above example the spatial resolution $r_A = A/(20m \times 20m)$, since the total measurement area is partitioned into sample sets of the aforementioned size. On the other hand, there is only one temporal sample set, as we calculate averages between 8-9am over all days in the dataset. Hence the temporal resolution r_T is just 1.

If the end-user instead wants to obtain a noise map for every 10 minutes in the specified hour, there would be 6 sample sets and so temporal resolution would be $1/6$ instead. Summarising the above reasoning, we see that our example campaign has achieved the desired quality if measurement density at time t, denoted $\rho(t)$, surpasses a particular prefixed value ρ_{min} (50 in the above). This can be rewritten in terms of resolution and total amount of campaign measurements at time t, $M(t)$, as follows.

$$\rho(t) \geq \rho_{min} \iff \frac{M(t)}{r_T.r_A} \geq \frac{M_{min}}{r_T.r_A} \quad . \tag{1}$$

More generally, resolution r_{s_i} for an arbitrary sensor stream s_i is the number of sample sets the corresponding predicate needs to be partitioned in, that is

$$r_{s_i} = \frac{|P_{s_i}|}{|\text{samplesets}_i|} \, . \tag{2}$$

Here vertical bars denote the *size* of a predicate, i.e. the size of the set it is defined by. Indeed the notion of resolution applies more generally with respect to arbitrary campaign protocols as defined in the previous section. For example, suppose we are interested in sound levels between 50 and 70 dB(A), in bands of 5 dB(A) each, we would need to include a resolution $r_{L_{eq}} = 4$. Similarly, focusing on a fixed set of tags {ambulance, truck, car} we would have a resolution $r_{tags} = 3$, as we would need to partition all valid campaign measurements over sample sets of size 1. Hence the general expression for campaign quality is as follows

$$P_r \equiv \rho(t) \geq \rho_{min} \iff \frac{M(t)}{r} \geq \frac{M_{min}}{r} \quad , \tag{3}$$

with the measurement resolution $r = \prod_i r_{s_i}$. Actually, the above expression is another predicate, which we call the *quality predicate* P_r, which forms an essential ingredient of a campaign protocol. Hence we need to adapt our definition as follows.

DEFINITION 2. *A* campaign protocol *is a triple of concatenated predicates $P_\Gamma + P_M + P_r$ over context Γ, measurements M and with measurement resolution r respectively.*

As we shall see below, checking the quality of a campaign at design time is one of the ways we put this notion to use, the other one being as a means to monitor a campaign's progress dynamically. Note that once a campaign protocol is fixed one could easily simplify this expression to $M(t) \geq M_{min}(t)$.

Which form of the predicate to use depends on the context: at design time the user-friendly density expression is more suitable, while at monitoring time the system can easily reason in terms of the total number of measurements.

CAMPAIGN LIFECYCLE

A campaign's lifecycle proceeds through a number of stages. First, a campaign protocol is *defined*. After the campaign is initiated, it is *monitored* to ensure it runs as intended. If this is not the case, *orchestration* procedures may be activated that enable to put the campaign back on track. Finally, after the campaign ends its outcomes are *analysed* and presented to interested parties. Before describing the implementation of our framework in the next section, we specify which role campaign protocol and predicates play in each element in a campaign's lifecycle. We note that in our past experiences with noise mapping campaigns, these campaign-specific coordination steps were carried out with very little ICT support. Rather, the workflow for running successful campaigns was partly a co-design process with the communities involved, and partly a research exercise in itself [6].

First of all a campaign needs to be defined, i.e. end-users need to be able to specify values of interest for each of the sensor streams in the PS setup the campaign runs in, as well as for the context in which these measurements are collected (by whom and in what way). In practically all cases this will include specifying an area of interest (e.g. by selecting an area on a map), often complemented by a time interval of interest. While the campaign definition tool should be general enough to be adaptable to various settings, it is also important that end-users need only focus on those streams that they are interested in. For example, grassroots communities wishing to map the noise in their neighbourhood are often not interested in specific tags, and thus should not have to detail any specifics w.r.t. this data stream. Another recurring element in our interactions with stakeholders is the possibility of dictating exactly who will contribute in a particular campaign rather than building up a volunteer base for a particular campaign organically. While end-users specify the above constraints by means of a user-friendly format (graphical, form-based...) internally these are translated precisely into predicates that form the campaign protocol. For example *private* and *public* campaign settings are nothing but particular context predicates P_Γ. These predicates are handed over to the monitoring framework in order to analyse which measurements are relevant to a campaign and which orchestration procedures need to be called in action if the campaign does not run according to plan. Knowing which predicate fails and at what rate is crucial in determining the remedial action required.

Campaign quality is the notion that binds all levels of a campaign's lifecycle. First of all, the soundness of a campaign protocol can be indicated at definition time, thus providing valuable feedback to campaign designers that can guide them in properly fine-tuning the protocol in advance. This is extremely useful for campaign managers knowledgeable on data quality issues, let alone for those that are not. To check protocol soundness at design time we use a statical version of the data quality predicate given in Eq.(3) to estimate the amount of measurements that can be gathered in the course of a campaign. Default values for resolutions can be used, while allowing more advanced users to specify their own values. Concretely, supposing measurements are gathered by n volunteers at a rate of R measurements per second (specific to the PS platform at hand). Then the maximal amount measurements M_{max} gathered during the campaign is given by

$$M_{max} = n.R.|P_{time}| \quad . \tag{4}$$

For the example campaign protocol mentioned earlier, assuming the set of employees consists of 10 people and a measurement rate of 1 measurement per second, we find $M_{max} = 1,116,000$. With a total area of $32.61km^2$ and default resolutions as above (i.e. one map for the whole period, divided in grid cells of $20m \times 20m$) we obtain a maximum measurement density of $\rho_{max} = 13$ measurements per grid cell. As this is below the desired minimum density $\rho_{min} = 50$ the system can then suggest the campaign designer to adapt the protocol, e.g. by increasing the number of volunteers or the duration of the campaign, or by decreasing the target area. Obviously there are more refined ways of making this estimate, e.g. by ignoring grid cells in unreachable areas.

Eq.(4) requires some caveats. First, we are assuming an identical rate for all sensors, but of course this may be something more complicated for a particular sensing setting. Second, we assume the number n of volunteers contributing is known in advance, constant, and that volunteers carry out the protocol exactly. Of course this is a highly optimistic assumption: in reality volunteers often contribute less than these maximal values, because of technological issues, localisation issues, or because they do not follow the protocol exactly, be it deliberately or not. On top of this, in public and even in private campaigns the number of volunteers n may evolve over time. Nevertheless, where possible one should estimate campaign quality in advance, as a first step towards completing a campaign successfully.

The fact that campaign protocols as specified at definition time may be executed quite differently as the campaign is running brings us to the *monitoring* phase. Here one essentially relies on the predicates in a campaign protocol to check which measurements are relevant to a campaign and which are not, at the same time maintaining a count $M(t)$ of said measurements. Note there is a difference in flavour (reflected in the implementation) between the contextual predicate P_Γ, which is defined on measurement tracks, and the predicate P_M, which is defined on individual measurements. The measurement count is used to check the campaign quality predicate dynamically. That is to say, after discrete time intervals (e.g. one hour) the measurement count $M(t)$ can be used instead of M_{max} to extrapolate whether campaign goals can be reached in terms of the protocol that is defined. If that is not the case, we can set into action a number of *orchestration* procedures, arguably the most useful aspect of a campaign framework such as the one we propose here.

In practice, campaigns almost always do not run as planned and without adequate support a lot of opportunities are missed

to carry out remedial actions. Indeed, in our personal experience it was very difficult to act upon incoming data in real-time, simply because the amount of work in analysing the incoming data, concretising the issues at the origin of divergent campaign evolution, and finally contacting the volunteers involved, is not manageable. By contrast, many of these elements can be automated, in particular because of our approach with campaign predicates. Indeed, knowing which predicate fails allows the system to identify the actual issue at hand and provide specific feedback to contributors (e.g. absent localisation data, measuring outside the area and time of the campaign, and so on). Also, by notifying the campaign designer of these issues, the protocol can be adapted in real-time so as to achieve data quality without disturbing the continuity of the campaign. All of these issues are highly important in an area which is gaining definite traction, but where motivation and engagement are crucial driving factors in maintaining incoming measurement streams. Unsuccessful campaigns can have a detrimental effect on first-time volunteers, and volunteers lie at the basis of any PS framework.

We now proceed with proposing an implementation of a framework for campaign management based on the design proposed in the above. Our framework system consists of two parts: the *campaign definition framework*, which enables users to specify and check the soundness of a campaign protocol, and the *campaign monitoring framework*, which monitors, and, if required, orchestrates running campaigns in terms of the protocol specified. For the campaign analysis phase we rely on techniques elaborated in earlier work [6], considering the development of more advanced or customisable analysis components as future work. The framework we present is elaborated on top of the existing NoiseTube platform [19], i.e. campaign abstractions are concretised as per the examples provided throughout this article. However, the abstractions and as such our implementation can be generalised towards other PS platforms, an endeavour which we will embark on in the near future. We state explicitly whenever our implementation is specific w.r.t. NoiseTube.

DEFINITION FRAMEWORK

To enable citizens to define their own campaigns, our system provides a user-friendly (web) interface for specifying the campaign protocol as given in Definition 2. We first explain how each of the predicates in a protocol are taken into account in our definition framework.

Context. The predicate P_Γ captures the desired context of the data collection, i.e. the particular type of users and/or devices that should be used for collecting campaign measurements. In our campaigning experiences we encountered two typical contextual specifications. First, campaigns can be declared as *private*, such that only trusted users (who can join through invitation) are allowed to participate. In other words P_u is defined as $\texttt{user} \in \texttt{U}$, with $\texttt{U}$ some predefined set. Conversely, a *public* campaign imposes no constraints on participants whatsoever, so that $P_u = \top$. Second, the issue of calibration of the sensors involved can be an important factor, as it strongly affects their accuracy. For example in NoiseTube not all compatible smartphone models have been calibrated, as it is very difficult to keep up with the market. Campaigns can thus make the choice of only including calibrated data, typically at the cost of decreasing the amount of relevant incoming data. Here the predicate P_d would be defined as in the example above, $\texttt{device} \in \{\texttt{calibrated-devices}\}$.[1]

At the moment, our campaign definition framework only considers contextual predicates P_Γ which are a combination of the predicates described above. While these contexts are likely to remain relevant for other PS frameworks, one could easily imagine more general contextual predicates such as gender and age to be important for stakeholders as well. Generalising towards arbitrary P_Γ is thus next on the list for framework improvements.

Measurements. Recall that P_M generally includes a temporal predicate P_T, alongside a geographical predicate P_A. Again, our framework focuses on these measurement predicates as we experienced them to be typical in campaigns for noise mapping and beyond. We also allow the possibility for campaign designers to define a tag predicate P_{tag}, which defines tags specific to the campaign at hand. Generalising towards other sensor data streams is another improvement planned for the near future.

The temporal constraint P_T can be specified by indicating the time and days which are of interest to the campaign. Both for days and for hours custom specification as well as predefined options are available. For days predefined options are full week, weekdays and weekend, while for hours one can focus on peak hours, non-peak hours, daytime and nighttime. There are two ways to specify a geographical constraint P_A in our framework: drawing a zone or a trajectory on a map. If a zone is selected, only measurements made in that zone are added to the campaign. In the case of a trajectory, every measurement made in a buffer of twenty meters around the trajectory is relevant for the campaign.

Data quality. The last step in the campaign definition process provides feedback with regards to the (static) data quality predicate P_r for the campaign specified. This feedback informs the campaign designer of the workload involved (in particular the number of participants required) to complete the campaign successfully in terms of the concrete constraints provided. This feature is extremely useful as it provides immediate feedback in case designers define unrealistic campaigns. For example, it will tell a campaign designer that creating a good map of a whole city involving only 10 participants for an hour per day during a week is simply not reasonable. Concretely, we compute the theoretical maximal amount of measurements collected according to Equation (4). Relying on default settings for temporal resolutions as exemplified above, we then check whether the associated ρ_{max} surpasses the value 50. Here we also rely on the measurement rate for NoiseTube. One obvious extension of our framework would be to allow user-specified resolutions and to include platform-specific rates. While the former option has not arisen in the context of campaigns executed, our inter-

[1] A list of calibrated phone models for NoiseTube can be found at http://www.noisetube.net/download/.

actions with more advanced, typically institutional stakeholders indicate that there is interest in opening up this feature.

Once a campaign is created, the framework enables members of the campaign (in addition to other users of the system, if the campaign is public) to view and discuss the campaign's progress by using a web-based interface. More details and screenshots can be found in [26, Chapter 4].

The campaign definition framework is built using the Ruby on Rails framework and is written as a combination of Ruby, HTML, CSS and JavaScript (as is the NoiseTube platform). In the backend sits a PostgreSQL database with the PostGIS spatial extension. We specifically chose the PostgreSQL DBMS in order to use PostGIS, since it adds data type support for geographic objects and allows various types of spatial queries to be handled at the SQL level.

MONITORING & ORCHESTRATION FRAMEWORK

To assist citizens to follow up on the enactment of their campaigns, we developed a monitoring framework that automatically checks incoming measurements for campaign relevancy and orchestrates remedial actions if the campaign does not evolve as intended. The monitoring framework relies on a number of repeatable patterns: for each campaign one has to check the context of each incoming track, the relevancy of measurements contained in this track, and the overall density of accumulated measurements. These repeatable patterns can be described as a *workflow*: a series of operations (called *activities*) which are enacted in a predefined order (specified by a defined set of rules) by a workflow engine. Workflows are particularly well-suited for modelling, coordinating and controlling various processes [9], and thus the underlying technology of choice. In the background section below we motivate the choice of workflow language in order to accommodate orchestration in the context of PS.

Background

PS systems are inherently distributed: a fixed infrastructure is complemented by flocks of mobile phone users, a setting known as a *nomadic network* [16]. Although a multitude of workflow languages are available nowadays [1], only a few focus on distributed network settings. Additionally, two characteristics of PS need to be taken into account when deciding upon which workflow language to use. First, in PS systems, participants use mobile devices (which act as services) to upload data to and receive feedback from a central server through unreliable wireless communication. As such, the workflow language should ensure execution progress in the face of unreliable communication and participants. Second, campaigns require a workflow engine capable of coordinating the execution of a group of participants' mobile applications. This is closely related to the concept of *group orchestration* [20] where a particular process (i.e. application) is executed for a group of services (e.g. users). Based on these two characteristics, we employ the nomadic workflow language called NOW [21]. To the best of our knowledge, NOW is the only workflow language which incorporates communication primitives resilient to network failures and has dedicated support for service and group orchestration. In what follows, we detail the necessary concepts of NOW to understand our campaign monitoring framework. More details on NOW can be found in [21].

Group orchestration. To manage a set of services (users) that form a logical group and coordinate the execution of processes for all group members, NOW introduces a new set of workflow patterns for group orchestration. The description of the services that belong to a group can either be achieved by enumerating all of them (extensional description) or by describing all properties those services must fulfil (intensional description). When a group pattern is instantiated, all the services satisfying the group's description are added to the group. Specially relevant in the context of PS is a *living group* pattern which allows to add newly discovered devices satisfying the group description dynamically after the start of the group pattern.

Monitoring & Orchestration Workflow

Our monitoring framework employs a workflow engine at the server backend which monitors the different campaigns. Each campaign thus has its dedicated workflow instance running on the server for as long as the campaign is active. Such a workflow is composed of three main services, as is depicted in Figure 1. *Track services* represent a participant's track. Everytime a user uploads a track, a track service is initialised. The second kind of service, called the *processor service*, provides a set of operations to analyse measurements in uploaded tracks in terms of campaign relevancy. Finally, the *monitor service* keeps track of the overall campaign progress. There is a 1-1 correspondence between these three types of services and the predicates composing a campaign protocol, as we explain below.

Context. Checking the context P_Γ is an operation at the level of tracks which requires group orchestration, i.e. for each uploaded track a particular process must be executed. The group demands an intensional description, which abstracts away from the precise number of tracks being uploaded during an active campaign. To do so, the workflow employs a group pattern as depicted by the rectangle with dashed lines at the top of Figure 1. By using the aforementioned living group pattern, we ensure that the workflow execution remains active for as long as the campaign is running. Each time a track service is uploaded, and hence discovered by the campaign workflow, the workflow engine verifies whether the track matches the intensional description. If this is the case, a new instance of the *measurement workflow* (the process wrapped by the group pattern) is instantiated for the track under consideration.

As a concrete example, consider a private campaign in which only invited/subscribed users can participate. When a track is uploaded by a participant of such a campaign and the description is satisfied, a new instance of the measurement workflow is initialised in which that particular track is processed. In order to reason about the context predicate P_Γ in a campaign protocol, NOW employs a reasoning engine integrated in the language [18]. The code necessary in NOWfor implementing a private campaign is shown in Listing 1. The intensional

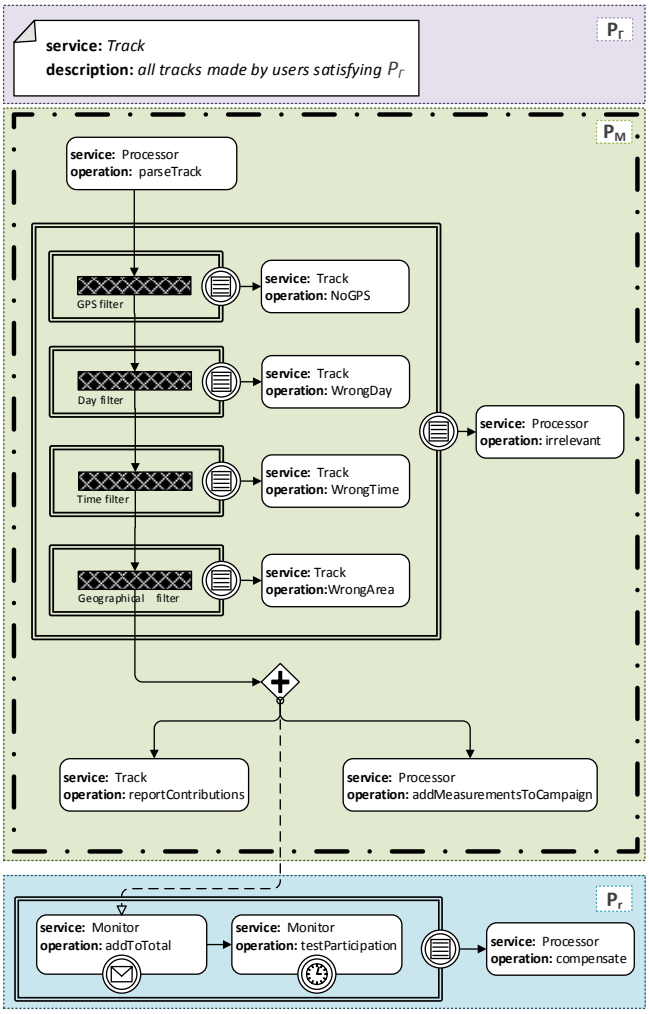

Figure 1. The campaign monitoring & orchestration workflow.

description of a private campaign consists of two prerequisites. The first prerequisite (line 1) is simply used to bind the variable `?CId` to the campaign that the workflow is representing. The second prerequisite (line 2) is used to test if the user who uploaded the track is a participant in a campaign. Both prerequisites are then combined (line 3) into an intensional description that reads as "all tracks made by users who are participants of this campaign". Lines 5-6 ensure that the measurement workflow is wrapped in the living group pattern as defined by its intentional description.

```
1  def p1 := private.is_campaign(?C, ?CId)
2  def p2 := shared.is_participant(?T, ?CId)
3  def desc := makeIntensionalDescr(p1, p2)
4
5  def campaignOrchestrator :=
6     LivingGroup(desc, 'track, MeasWF)
```

Listing 1. Rules to implement context P_Γ for a private campaign.

Measurements. The measurement workflow (depicted within the black dotted lines) is executed only for tracks that satisfied the aforementioned group's intensional description, i.e. the tracks that fulfil the context P_Γ. It relies on the processor service to verify whether the individual measurements of a track also satisfy the measurement predicate P_M imposed by the campaign protocol. *Measurement verification* is realised by parsing the track such that its measurements are available during the process' execution. The core of the verification process is implemented using a series of filter patterns, one for each of the components of P_M. Recall that P_M generally includes temporal and geographical predicates. The temporal predicate is enforced by the *day filter* and the *time filter*, which filters out measurements made on the wrong day or at the wrong time respectively. The geographical predicate is enforced by the *GPS filter*, which checks if location data is available, and the *geographical filter*, which verifies whether the measurements are made in the right area (zone or trajectory). Here we make heavy use of the PostGIS spatial extension of our database, as it greatly facilitates this sort of spatial queries. Note that any of these filters could be absent from the workflow, as the constraints they enforce are optional.

Each filter is surrounded by a *constraint violation pattern* (depicted by the double rectangle labeled with a sticky note symbol) which monitors the amount of measurements that are filtered out at each step. Constraint violation patterns can trigger compensating actions which are the engine behind campaign orchestration. These actions do not stop the execution of the workflow, which continues processing measurements, but is used to provide intermediary feedback to participants. This filter-specific feedback is crucial for participants to enable them to correct their measurement actions. For example, consider a campaign that requires participants to follow a predefined trajectory. If some participant accidentally diverts from the trajectory, we can inform him/her of this issue so that it can be corrected for. Moreover, there are indications that feedback of any kind sustains participants' motivation to contribute. Our framework's feedback system is currently text-based (e.g. "56 measurements of this track were made in the wrong area"). However, it is perfectly feasible to implement more user-friendly approaches such as a graphical representation showing a participant's track and using colour coding to indicate where s/he diverted from the trajectory.

After all measurements in a track are processed, the verification process is said to be finished. The user is then informed about how many measurements contributed to the campaign (as shown by the `reportContributions` operation), and these measurements are added to the campaign by the `addMeasurementsToCampaign` operation. This basically consists of linking the measurements in the underlying database and making them visible on the campaign webpage. Additionally, a signal is sent to the campaign quality monitoring workflow discussed below.

Quality. The campaign monitoring workflow, depicted at the bottom of Figure 1, is responsible for monitoring overall campaign progress in terms of the predicate P_r. It is worth noting that this part is separated from the measurement workflow since campaign quality is a notion specific to a campaign rather than individual tracks. This workflow relies on the monitor service which is notified each time relevant measurements are added and keeps track of the total number of campaign measurements $M(t)$.

In parallel, a timer mechanism periodically tests the campaign progress. Currently this mechanism proceeds on an hourly basis, first checking whether the hour that just passed was of interest to the campaign. If it was not, no further calculations are performed. In case the hour is included in the campaign protocol, the activity first verifies whether P_r holds, i.e. whether enough measurements have been accumulated to terminate the campaign. If P_r is not yet satisfied, the current value of $M(t)$ is extrapolated for the campaign duration according to the procedure described earlier. Using this information, we can inform the campaign manager on the required runtime for the campaign for achieving P_r following the current average contribution rate. In order to give feedback when the number of measurements added each hour is lower than the required average, the workflow uses another constraint violation pattern. The compensating action for this pattern posts a message on that campaign webpage notifying participants that a higher participation grade is requested. Alternatively, the campaign manager can be contacted to advice him/her to increase the runtime of the campaign or the number of participants so that the data quality of the campaign is satisfied.

VALIDATION

We now validate our approach in terms of correctness of campaign outcomes and efficiency of running campaigns. We do this by comparing these aspects for 6 noise mapping campaigns with and without the availability of our campaign framework. These campaigns were first *manually* executed by a platform owner who served the role of campaign designer, monitor and analyser. We have then re-enacted (i.e. simulated) them using our framework for *automatic* campaign management which takes up and extends upon those same roles. By comparing both processes for each of the 6 campaigns we quantify how the campaign final outcome is affected (correctness), how campaign execution can be improved (efficiency), and, finally, how the societal uptake of campaigning can be boosted by providing support for more autonomous community campaigning (scalability).

Before continuing, we briefly describe these 6 campaigns. Each campaign followed a step-wise process, where face-to-face meetings interleaved with intermediary bespoke developments for campaign design, definition, monitoring and analysis respectively. The campaigns were initiated by a variety of stakeholders (including researchers, grassroots organisations, city administrations, and within an educational context).[2]. All campaigns focused on mapping noise pollution, one campaign also mapping air pollution and meteorological parameters. In the Wommelgem (WO) campaign, 7 concerned citizens mapped the noise pollution in their neighbourhood due to a huge traffic-laden roundabout. The Baia Mare (BA) campaign noise mapped a 3-km path through the town historical center with 6 volunteers. The campaign in Zagreb (ZA) was coordinated with the city administration which recruited 5 volunteers to follow a predefined trajectory during peak hours. The Linkeroever (LI) campaign tasked 8 volunteers to measure along a predefined path in their neighbourhood. The BxlSense campaign held in the city centre of Brussels (BX) focused on mapping noise and other environmental data using off-board sensors. Finally, in Tuinwijk (TU), up to 19 volunteers measured noise for at least 1 hour a day while walking around in a predefined zone. Further details about these campaigns can be found in [26].

[2]The campaign maps are available at `www.brussense.be/experiments`

	WO	BA	ZA	LI	BX	TU
with	56239	16585	44453	26713	21709	24429
w/o	66645	22045	59162	35600	31923	23448
%	16	25	25	25	32	-4

Table 1. Relevant measurements with and without the framework.

Correctness

The first aspect to check is whether our framework produces correct outcomes. *Correctness* in this context means that the maps produced are based only on measurements that satisfy the campaign protocol, and that statistics on the composing sample sets are correctly computed. The statistical analysis of measurements relies on the procedure proposed in [6], and has been incorporated unchanged into the framework.

Table 1 shows the number of relevant measurements identified with and without the framework in place for the 6 aforementioned campaigns. Relevant measurements correspond to those measurements that satisfy the campaign protocol. In most cases, more measurements were identified to be relevant without the framework than with (except for the Tuinwijk campaign). Closer inspection of the datasets reveals that the difference in relevant measurements is partly due to human errors. In the case of the Tuinwijk campaign, an unexpected user was contributing data. This was caught by our framework but not by the platform owner. For the Zagreb campaign, measurements of up to two hours beyond the limits of the campaign protocol were wrongly included. However, in this case, the platform owner manifested that some wrong measurements were deliberately incorporated as a way of increasing campaign size by softening protocol boundaries. For example, measurements taken 10 minutes before and after the temporal predicate were tolerated. This shows that our framework is definitely valuable in catching errors, but it is still worthwhile to keep a human involved in the campaign management, e.g. to soften protocol boundaries if necessary.

Efficiency

We now evaluate how our framework improves upon efficiency in terms of campaign *runtime*, i.e. the time that a campaign runs. To this end, we compare the runtime of 6 campaigns with the potential decreased runtimes if its participants would have received feedback on the collected data. Our framework provides feedback for three types of data collection errors: (1) missing localisation data, (2) data collection at times/days that are not relevant, and (3) data collection outside the designated measurement area. As we are re-enacting past campaigns, we need to make some assumptions on how the participants will react to the feedback they receive.

	total	error type			relevant
	meas.	GPS	time	area	meas.
Track 1-real	2731	744	116	270	1601
Track 2-real	2162	248	319	240	1356
Track 2-learned	2162	0	0	240	1923

Table 2. Comparison of campaign tracks with and without learning.

We assume that there is an 80% chance that a user corrects for a measurement error, and this for each type of error that receives feedback on. Once a user learns how to prevent a particular error, we consider all future measurements (previously unusable due to the error) as relevant. If a user did not correct the errors, this decision is passed on to the next track, where the process is repeated. All errors need to be corrected for a measurement to be relevant. Note that missing localisation data and measurements at irrelevant areas are mutually exclusive errors.

Table 2 lists two consecutive tracks by one particular participant, which we use as an example to explain our methodology. The first track consists of 2731 real measurements of which 1601 are relevant to the campaign. The remaining 1130 measurements contain an error, and are split in the table according to the three aforementioned errors. For each error, this user has a 80% chance not to make it in the future. The third line in the table shows one particular outcome for the second track in which the user learns not to make the first two types of errors again. The real measurements for the second track are thus adapted in the third line, which contains 1923 usable measurements instead of 1356.

For each campaign we use the actual running time and the associated total amount of usable measurements as the reference point for our evaluation. Using the above approach, we check at what point in time in the campaign's re-enactment we reach the same amount of measurements. To account for nondeterminism due to the 80% learning rule, we take the average of 10 simulations per user. Figure 2 depicts the outcome of this analysis, comparing the actual runtime of these campaign (in days) to the runtime these campaigns would have had when benefiting from automated orchestration. We find decreased runtimes of 16% (Zagreb) up to 50% (Wommelgem). As campaign managers this difference can be very clearly coupled to the organisation of the volunteer groups involved and their technical expertise, which was much better in the case of the former than the latter. While the results vary, and of course depend on the learning assumptions used, it should be clear that the presence of the orchestration framework has a definite impact on campaign efficiency.

Scalability

The above clearly shows that our current implementation improves campaign runtimes. To scale up also in terms of the *number* of campaigns we investigate whether our design enables the full campaign lifecycle to be carried more efficiently. To this end, we estimate the effort spent in organising 19 campaigns from first-hand experiences, identifying those

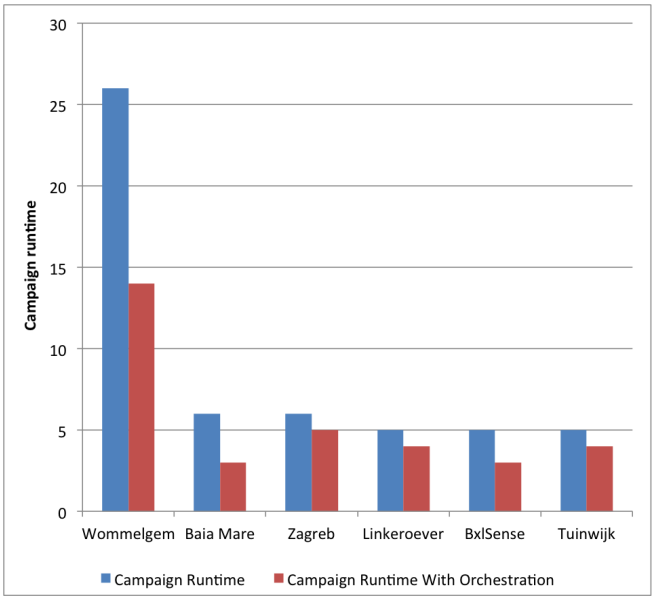

Figure 2. Campaign runtime with and without orchestration.

elements which can be handled by the framework. By involving platform owners to a much lesser extent, communities are empowered to run campaigns more autonomously.

The analysis presented here is tentative in several ways. First, it assumes a framework fully implementing the design proposed in the earlier sections of this paper (in particular sensor abstractions), which moreover adequately disseminates best practices guidelines (in the form of manuals, copyable campaign designs, ...). Second, we did not carry out an in-depth study of the efficiency of our framework in other application areas and with real communities, which we consider beyond the scope of this paper. Rather, we relied on our own experiences as campaign designers, managers and analysers, which we believe to be valuable nonetheless. Third, we did not consider the benefits of dynamical monitoring the quality predicate, which enables campaign managers to adapt the protocol dynamically.

The outcome of our analysis in guiding 19 campaigns is summarised in Table 3. The table estimates the time spent on campaign design and management with and without our platform according to three types of campaigns: (1) the very first noise mapping campaign described in [6], (2) *copycat campaigns*, i.e. campaigns following a similar approach as the first campaign (a campaign in a predefined area during peak/off hours), and (3) small variation campaigns. The latter category is represented by the BxlSense campaign which, as previously mentioned, focused not only on noise but also on environmental measurements with offboard sensors. The first column of Table 3 estimates the time spent in days for the combined steps in a campaign's lifecycle without the platform in place. To this end, we examined agendas, emails, in-house time sheets, file histories and so on. Depending on the deviation of a campaign with respect to earlier campaigns, campaign management was more time-consuming or not. For example, configuring the technological platform for campaign-

Campaign type	w/o platform	with platform	effort saved
1st noise mapping campaign [6]	24 + O	8.5	15.5 + O
copycat campaigns	13 + O	2	11 + O
small variations	34 + O	13.5	20.5 + O

Table 3. Estimations of campaign management effort without and with our platform (in days), with O for orchestration effort.

specific data collection took on average 2 days for a copycat campaign, but 2 weeks for the first noise mapping campaign and 4 weeks for the first environmental mapping campaign. These differences indicate the importance of developing campaign frameworks that are geared towards reconfigurability, to avoid the considerable bespoke software engineering that is an important obstacle to wide-scale adoption of PS techniques.

It is important to realise that the effort calculated does not include dynamic monitoring and orchestration, as this was simply not feasible with the staff and timeframes at hand. Indeed, analysing data streams and getting back to individual volunteers is considerable unwieldy when no ICT support is present, involving manual data inspection and direct interaction with volunteers. If the protocol is not followed or, worse, technical or usability issues prevent the measurements from arriving at the system at all, it requires a substantial amount of time on problem diagnosis and resolution. Because of this limitation, in the actual campaigns run we limited orchestration to resolving issues at the end of each campaign data collection phase. Denoting the effort for manual monitoring and orchestration (i.e. without framework support) as O, we thus need to include this quantity specifically in calculated campaign efforts to obtain the full picture. While this quantity depends highly on the campaign protocol and how it is executed, the effort required should not be underestimated and is indeed one of the main reasons for developing the implementation presented here. Without monitoring, campaigns do not progress as efficiently as Figure 2 underlines.

The second column of Table 3 shows the estimated total efforts on campaign design and management with the platform in place. To generate these estimates, we analysed which steps in the process can be taken up by the platform. We find that effort could be decreased at all levels of the campaign lifecycle: campaign protocol design, analysis and orchestration is substantially facilitated by the implementation in place, while best practices guidelines allow to decrease the number and duration of interactions of platform owners and campaign managers. In terms of bespoke software engineering efforts it is more difficult to quantify the efficiency gained. For the sake of simplicity we assume here that 50% of the effort could be saved by working with a framework implementing the abstractions proposed in the above, and thus easy to reconfigure to varied settings. Communities using the platform should share a reasonable level of autonomy to start with, such as being able to use their own measurement devices and/or borrow ours without much intervention from platform owners.

The last column of Table 3 summarises the estimated effort gained by using the platform, writing again O for orchestration effort.

From our analysis of the 19 campaigns that we have manually managed as platform owners, we believe the effort for new campaigns could be cut by more than half, and the effort for copycat campaigns could be reduced even more significantly. These numbers are highly supportive of our platform. Validations in terms of running campaigns involving real communities are needed to obtain more accurate numbers on the actual improvements achieved.

CONCLUSION & FUTURE WORK

At the heart of this article lies the notion of a participatory sensing campaign. We argue the need for ICT support to achieve scalability over the full campaign lifecycle, and for orchestration support to achieve campaign quality. After this we propose relevant campaign abstractions relying on first-hand experience with more than a dozen campaigns in the sound measuring domain and on the state of the art in PS. We then use these abstractions as the basis of the design and implementation of a framework for campaign management, relying on workflow technology and meta-logic reasoning. While our implementation is based on an existing platform for participatory noise monitoring, the underlying abstractions are phrased with arbitrary PS settings in mind. To our knowledge, this is the first time campaigns and a platform for supporting them have been so thoroughly studied, developed and implemented. We also present a validation of our approach by comparing campaigns previously executed without the platform with a re-enactment thereof supported by the platform, quantifying the increase in correctness, efficiency and scalability.

There are, as always, many avenues for further research. In the short run, there are several features that would improve our system as is, such as supporting the full spectrum of protocol predicates, heterogeneous data monitoring and user-defined campaign quality. This should be complemented by a validation of in terms of real-time community-based campaigning activities, which in turn should point towards further improvements. A larger — and more important — effort is to do with upscaling the system towards more users, more campaigns, and more parameters, so that eventually we have a fully reconfigurable system able to answer variable stakeholder requirements. Though this will certainly prove to be a much more substantial research and engineering effort, we believe that by the approach presented here we have optimised our chances for success. Finally, one should always bear in mind that end-users are typically non-ICT experts, and thus the abstractions presented should serve as a backbone for a user interface which is geared towards a variety of communities and stakeholders. Our experience in running campaigns and frequent contacts with stakeholders at the local, regional, national and international level will be an essential asset here.

ACKNOWLEDGMENTS

This work has been supported by Innoviris and the EU-CIP i-Scope project. We wish to thank all citizens and societal stakeholders who have directly or indirectly contributed to this work.

REFERENCES

1. Barker, A., and Hemert, J. Scientific workflow: A survey and research directions. In *Parallel Processing and Applied Mathematics*, R. Wyrzykowski, J. Dongarra, K. Karczewski, and J. Wasniewski, Eds., vol. 4967 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2008, 746–753.
2. Boulos, M. N. K., Resch, B., Crowley, D. N., Breslin, J. G., Sohn, G., Burtner, R., Pike, W. A., Jezierski, E., and Chuang, K.-Y. S. Crowdsourcing, citizen sensing and sensor web technologies for public and environmental health surveillance and crisis management: trends, OGC standards and application examples. *International journal of health geographics 10*, 1 (2011), 67.
3. David M. Aanensen, Derek M. Huntley, Edward J. Feil, Fada'a al Own, and Brian G. Spratt. EpiCollect: Linking Smartphones to Web Applications for Epidemiology, Ecology and Community Data Collection. *PLoS ONE 4*, 9 (2009).
4. Delphine Christin, Andreas Reinhardt, Salil S. Kanhere, and Matthias Hollick. A survey on privacy in mobile participatory sensing applications. *Journal of Systems and Software 84*, 11 (2011), 1928–1946.
5. Eisenman, S. B., Miluzzo, E., Lane, N. D., Peterson, R. A., Ahn, G.-S., and Campbell, A. T. Bikenet: A mobile sensing system for cyclist experience mapping. *ACM Trans. Sen. Netw. 6*, 1 (2010), 6:1–6:39.
6. Ellie D'Hondt, Matthias Stevens, and An Jacobs. Participatory noise mapping works! An evaluation of participatory sensing as an alternative to standard techniques for environmental monitoring. *Pervasive and Mobile Computing 9*, 5 (2013), 681–694.
7. Eric Paulos. Citizen Science: Enabling Participatory Urbanism. In *Handbook of Research on Urban Informatics: The Practice and Promise of the Real-Time City*, M. Foth, Ed. Information Science Reference, IGI Global, 2009, ch. 28, 414–436.
8. Ganti, R. K., Pham, N., Ahmadi, H., Nangia, S., and Abdelzaher, T. F. Greengps: A participatory sensing fuel-efficient maps application. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, ACM (2010), 151–164.
9. Hollingsworth, D., and Hampshire, U. Workflow management coalition the workflow reference model. *Workflow Management Coalition* (1993), 68.
10. iSPEX. Measure aerosols with your smartphone, 2013. `http://ispex.nl/en`.
11. Jeffrey A. Burke, Deborah Estrin, Mark Hansen, Andrew Parker, Nithya Ramanathan, Sasank Reddy, and Mani B. Srivastava. Participatory Sensing. In *WSW'06: Workshop on World-Sensor-Web, held at ACM SenSys '06* (2006).
12. Kim, S., Mankoff, J., and Paulos, E. Sensr: Evaluating a flexible framework for authoring mobile data-collection tools for citizen science. In *Proc. Computer Supported Cooperative Work*, ACM (2013), 1453–1462.
13. Lane, N. D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., and Campbell, A. T. A survey of mobile phone sensing. *Comm. Mag. 48*, 9 (2010), 140–150.
14. Linnap, M., and Rice, A. The effectiveness of centralised management for reducing wasted effort in participatory sensing. In *First International Workshop on Crowdsensing Methods, Techniques, and Applications* (2014).
15. Maisonneuve, N., Stevens, M., and Ochab, B. Participatory noise pollution monitoring using mobile phones. *Information Polity 15*, 1 (2010), 51–71.
16. Mascolo, C., Capra, L., and Emmerich, W. Mobile computing middleware. In *Advanced Lectures on Networking, NETWORKING 2002*, Springer-Verlag (2002), 20–58.
17. Miluzzo, E., Lane, N., Eisenman, S., and Campbell, A. Cenceme – injecting sensing presence into social networking applications. In *Smart Sensing and Context*, G. Kortuem, J. Finney, R. Lea, and V. Sundramoorthy, Eds., vol. 4793 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2007, 1–28.
18. Mostinckx, S., Scholliers, C., Philips, E., Herzeel, C., and De Meuter, W. Fact spaces: Coordination in the face of disconnection. In *Proc. Coordination Models and Languages*, COORDINATION'07, Springer-Verlag (2007), 268–285.
19. Nicolas Maisonneuve, Matthias Stevens, and Bartek Ochab. Participatory noise pollution monitoring using mobile phones. *Information Polity 15*, 1-2 (2010), 51–71.
20. Philips, E. *Workflow Abstractions for Orchestrating Services in Nomadic Networks*. PhD thesis, Vrije Universiteit Brussel, 2013.
21. Philips, E., Straeten, R. V. D., and Jonckers, V. NOW: Orchestrating services in a nomadic network using a dedicated workflow language. *Sci. Comput. Program. 78*, 2 (2013), 168–194.
22. Ramanathan, N., Alquaddoomi, F., Falaki, H., George, D., Hsieh, C., Jenkins, J., Ketcham, C., Longstaff, B., Ooms, J., Selsky, J., Tangmunarunkit, H., and Estrin, D. ohmage: An open mobile system for activity and experience sampling. In *Proc . PervasiveHealth* (2012), 203–204.

23. Rula, J. P., and Bustamante, F. E. Crowd (soft) control: moving beyond the opportunistic. In *HotMobile*, G. Borriello and R. K. Balan, Eds., ACM (2012), 3.

24. Tilak, S. Real-world deployments of participatory sensing applications: Current trends and future directions. *ISRN Sensor Networks 2013* (2013).

25. von Kaenel, M., Sommer, P., and Wattenhofer, R. Ikarus: Large-scale participatory sensing at high altitudes. In *2th Workshop on Mobile Computing Systems and Applications (HotMobile)* (2011).

26. Zaman, J. Orchestrating participatory sensing campaigns with workflows. Master's thesis, Vrije Universiteit Brussel, 2013. `http://brussense.be/Pubs/files/Zaman_MScThesis_2013.pdf`.

27. Zaman, J., D'Hondt, E., Gonzalez Boix, E., Philips, E., Kambona, K., and De Meuter, W. Citizen-Friendly participatory campaign support. In *PerCom WiP'14* (2014), 248–251.