2012 8th IEEE International Conference on Distributed Computing in Sensor Systems

A Ubiquitous Publish/Subscribe Platform for Wireless Sensor Networks with Mobile Mules

Xiaoyu Tong and Edith C.-H. Ngai
Department of Information Technology
Uppsala University, Sweden
edith.ngai@it.uu.se

***Abstract*— Existing publish/subscribe architectures for wireless sensors networks only support stationary sensors interconnected with each other. They cannot handle mobile sensors or remote sensors that are sparsely deployed. In this paper, we propose a novel ubiquitous publish/subscribe system that supports data access from both mobile sensors and stationary sensors. Our system utilizes mobile phones as data mules to relay subscriptions and published data between broker and remote sensors. It provides content-based publish/subscribe services from sensors deployed anywhere without depending on any network infrastructure. We implement our publish/subscribe platform on real hardwares and test it in a hiking trail application. The application allows users to subscribe for sensing data from both stationary sensors and mobile sensors along hiking trails. Extensive experiments are conducted in an outdoor testbed to evaluate the system performances such data delay, number of data received and communication overhead.**

I. Introduction

During the last decade, wireless sensor networks (WSNs) [3] have attracted much attention from the research community. They are usually formed by a number of wireless sensors, which are utilized to collect sensing data from the environment [25], [16]. With the advanced mobile technology, smart phones become incredibly popular as new kind of sensing devices with their equipped camera, GPS, and accelerometer, etc. Mobile phones have stronger computation power, network connectivities, memory and batteries, which are ideal counterparts of stationary sensors for ubiquitous sensing.

Publish/subscribe paradigm has been recognized to foster decoupling of distributed objects in large-scale distributed systems. It can support large amount of information sharing among asynchronous devices like WSNs. In many publish/subscribe architectures, subscribers such as Internet users register their subscriptions with a broker in the network. Publishers such as sensors post their collected sensing data to the same broker. The broker, which acts as both server and database, performs filtering and distributes the subscribed data from the publishers to the subscribers. Although various publish/subscribe systems have been proposed for WSNs, existing architectures only target for WSNs with stationary and interconnected wireless sensors. Most of them rely on multi-hop routing to forward data to the brokers, which are not applicable for sparsely deployed wireless sensors. The situations become even worse in outdoor or remote areas, where network infrastructures are hardly available. Providing ubiquitous access to sensing data remains a challenge due to the constraints of network connectivities, availability of sensors, limited sensing coverage and capabilities, etc. The complexity of heterogeneous mobile sensing devices such as smart phones can further complicate the problem.

In this paper, we propose a novel publish/subscribe system which supports ubiquitous data access from both wireless sensors and mobile phones. It can provide content-based publish/subscribe with high level of abstraction from the underlying sensors and network infrastructures. Users can subscribe for sensing data by simply specifying the target area, sensing types and data ranges of interest without knowing the addresses or locations of the sensors. To enable data access from remote sensors, we suggest mobile phones to be utilized as mobile mules to relay subscriptions and published data between the broker and the wireless sensors. Mobile users can perform sensing using their phones and collect sensing data from the wireless sensors that they pass by. The deployment of WSNs becomes more flexible without relying on any additional network infrastructure. We implement the proposed platform on real hardwares and demonstrate successful ubiquitous publish/subscribe services for both wireless sensors and mobile phones.

We summarize the key contributions of this work here. (1) To the best of our knowledge, we are the first to propose and implement a ubiquitous publish/subscribe platform that supports heterogeneous sensing devices, including remotely deployed wireless sensors and mobile phones. (2) We suggest mobile phones to be utilized as both mobile sensors and mobile mules to relay subscriptions and published data between the broker and the wireless sensors. Adaptive location updates are proposed to reduce the communication overhead of mobile phones without degrading the publish/subscribe services. (3) We implement and experiment our platform in an outdoor sensor network testbed. A hiking trail application has been developed successfully on top of our publish/subscribe platform, which can provide subscribers with sensing data such as temperature, humidity, light intensity and hiking speeds measured by both wireless sensors and mobile phones. Extensive evaluations have been conducted to evaluate network performances including data delivery delay, number of data received and communication overhead.

The remaining of the paper is organized as follows. Section II discusses the related work on publish/subscribe for WSNs. Section III describes the network models and our design goals.

978-0-7695-4707-7/12 $26.00 © 2012 IEEE
DOI 10.1109/DCOSS.2012.65

We introduce our ubiquitous publish/subscribe platform for WSNs with mobile mules in Section IV. We describe the implementation of our proposed system on real hardwares in Section V. Section VI presents the experiment settings and evaluates the performance of our platform in a hiking trail application. Section VII concludes the paper and gives the future work.

II. Related Work

Publish/subscribe paradigms have been widely studied in peer-to-peer networks and wireless sensor networks. MQTT-S [14] is a publish/subscribe protocol, which employs a topic-based scheme for making subscriptions in WSNs. Its goals is to enable queries to sensor nodes directly from the Internet. Mires [23] is another publish/subscribe middleware for sensor networks, but it adopts a content-based approach by incorporating characteristics of message-oriented middleware. Shi et al. also proposed a content-based publish/subscribe middleware called TinyMQ [22]. In TinyMQ, an overlay network is constructed on top of the underlying WSN, in which sensor nodes can be logically connected independent of their geographical locations. However, all of the above architectures are designed only for stationary WSNs, which do not support mobile sensors and mobile mules.

Different implementations of publish/subscribe have been proposed for WSNs. Albano et al. [4] studied publish/subscribe in WSNs based on Data Centric Storage system. It supports sensing data storage and retrieval over a network layer providing simple unicast and broadcast primitives. Hauer et al. [12] further designed a content-based publish/subscribe framework applying a component-based architecture to simplify application developments. Developers are free to specify communication protocols, data attributes, and service extension components according to their needs. *Virtual brokers* have been proposed to support implementation of large-scale publish/subscribe for WSNs [17]. The idea is to distribute the functionalities of centralized brokers into a group of ordinary sensors by partitioning the connected network graph of the WSN. Different from the above work, we do not assume sensors nodes to be interconnected in our platform. Our platform can tackle limited network connectivity of remote sensors by utilizing mobile phones as data mules to relay the subscriptions and the published data. Moreover, it can support heterogeneous sensor networks with both stationary sensors and mobile phones.

By extending WSN services to the Internet, diverse sensing data could be accessed from users anytime and anywhere. SenseWeb [15] is a system aiming at facilitating data access from all the shared sensors across the entire Internet. With a tasking module in the coordinator component, this system optimizes sensor selection for different applications based on their requirements. Priyantha et al. [19] attempted to solve the scalability problem for WSNs by employing web services. It adopts TCP/IP stack and HTTP in sensor nodes to implement web services. Sensor nodes are accessible directly from the Internet through web services. Similarly, Aberer et al. [1] proposed a middleware called Global Sensor Networks (GSN), which provides a high level abstraction that eases the interconnection with WSNs. Although both SenseWeb and GSN used proxies for connecting WSNs to the Internet, their proxies are stationary and constantly connected to the WSNs, which are fundamentally different from our design. In our platform, mobile phones can act as data mules to relay subscriptions and sensing data from remote sensors in a delay tolerant fashion.

Apart from stationary networks, publish/subscribe for mobile environments have been explored. Huang et al. [13] described how to distribute the system across multiple computers or mobile devices to distribute load and cope with failures, message loss and disconnections. Fiege et al. [10] sketched how physical mobility and logical mobility can be implemented within the existing REBECA [6] event-based system. Publish/subscribe architectures have also been widely investigated in mobile ad hoc networks (MANETs) [20], [26]. Frey et al. [11] proposed a publish-subscribe paradigm that can manage and exploit context information when matching events against subscriptions. This allows publishers and subscribers to control their diffusion and to restrict the identities of their communication parties. Costa et al. proposed SocialCast [7] as a routing framework for publish-subscribe that exploits social interaction and mobility pattern of human beings to identify the best information carriers. Although mobility in publish/subscribe has been widely studied, publish/subscribe for mobile sensing environment has not been fully investigated. In particular, the sensing context, heterogeneity of sensing devices, mobility of nodes and limited connectivity of wireless sensors remain to be further explored.

III. Preliminaries

A. Network Models

In the rest of this paper, the stationary sensors in our system are referred as SSensors, while the mobile sensors on smart phones are referred as MSensors. Our platform intends to provide publish/subscribe services for Internet users to access data from both MSensors and SSensors. We present the SSensors, MSensors, mobile proxy, and consumers in our system from a network perspective in Figure 1.

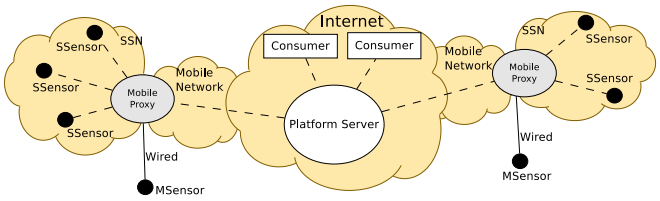

Fig. 1. Overview of our publish/subscribe system.

Platform server is responsible for relaying messages (e.g. subscriptions and published sensing data) between consumers and mobile proxies on the Internet. Similar to the platform server but in a lower level, mobile proxies relay messages between the platform server and the connected sensors. They are usually mobile phones that are utilized as data mules to

collect data from different devices, including sparse sensor networks (SSNs). The mobile phones can communication with SSensors through short-range communication, such as bluetooth or IEEE 802.15.4. Other that that, they can connect to the platform server through 3G or WiFi connectivity.

B. Design Goals

Our system aims at providing publish/subscribe services for both stationary sensors and mobile phones. It has to provide two basic functions to achieve this goal. The first function is to forward subscriptions from Internet users to the capable sensors (both SSensors and MSensors). Another function is to deliver the published sensing data from the sensors to the subscribers. Users should be able to subscribe for sensing data of interests without knowing the locations of the underlying sensors and the implementation details. The mobility and availability of the sensors should be transparent to the users. Since mobile sensors and stationary sensors usually have distinct sensing functions, users can obtain more diverse types of sensing data.

The platform is designed to support publish/subscribe services even for remotely or sparsely deployed wireless sensors, given that mobile users can move around and communicate with the SSensors opportunistically. This design greatly increases the flexibility of sensor deployment without relying on any existing network infrastructures. The mobile phones can provide intermittent connection between the platform server and the SSensors in a delay tolerant manner. For this reason, the communication between different devices in this system should be implemented as asynchronous as possible. For instance, when the platform server receives an subscription, it can be no mobile proxy available in the subscribed area at the moment. The subscription then has to be stored in the server for a period of time before successfully delivered to the sensors. Similarly, when the data are published by the sensors, they have to be cached at the sensors before any mobile proxy approaches to pick them up.

IV. Publish/Subscribe Platform Design

We outline the design of our publish/subscribe system in Figure 2. The system is developed based on a publish/subscribe architecture due to its advantage of fully decoupling in time, space and synchronization among different components in a distributed system. Since users may have diverse interests on the content of sensing data (e.g. temperatures above 20 degrees), our platform should have enough expressiveness to reflect their requests. As a result, the content-based publish/subscribe is adopted in this platform. Subscribers are the Internet or mobile phone users who request for sensing data of interest. Sensing data are collected by both the stationary and mobile sensors when events of interest are detected. In a publish/subscribe context, consumers in our platform are referred as subscribers, while the SSensors and MSensors are referred as publishers. The platform server is named as *broker*, which caches and relays messages between the subscribers and the publishers. Mobile phones are utilized as *mobile brokers* (MB) to relay data from stationary sensors to the broker.

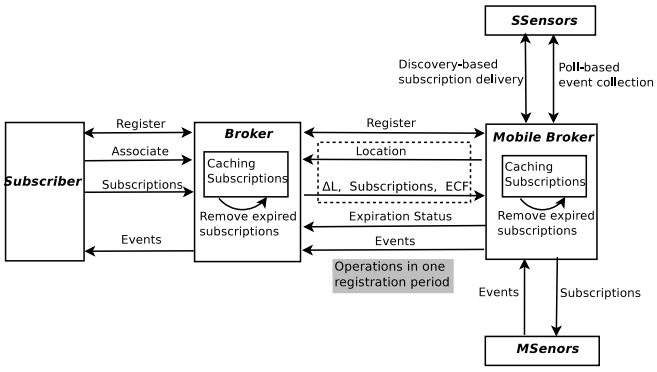

Fig. 2. Operations in our publish/subscribe platform.

A. Subscriber Registration

Internet or mobile users can subscribe to the sensor data through the broker. Each subscription is represented by a "SbSubscription" message including the "targetArea", "temporalRange" and "Constraints" fields (see Figure 3). The "targetArea" field indicates the geographical area of interest in the format of *targetArea* = {*point, radius*}, where "point" and 'radius" are the geographic coordinates of the center and radius of the area. Subscribers, who are interested in sensing events from a special area (e.g. temperatures in area A), can subscribe for their sensing data easily with the above format. The message also defines the starting time and ending time, as well as other specific constraints to the publish/subscribe services.

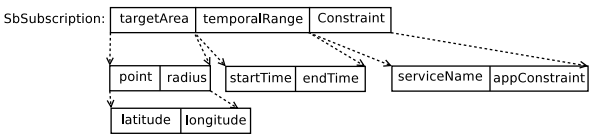

Fig. 3. Format of subscription message from subscriber to broker.

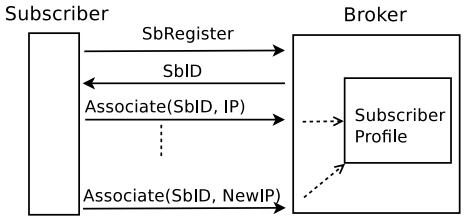

Fig. 4. Subscriber registration and association to the broker.

Before starting to use the platform, each subscriber registers itself to the platform server. This is performed by sending a "SbRegister" message to the broker as shown in Figure 4. Once receiving the message, the broker will allocate an unique "SbID" to the subscriber. At the same time, a blank

profile will be created for the subscriber. The subscriber can then associate itself to the broker by sending an "Associate" message. The parameters "SbID" and "IP" of the subscriber from the "Associate" message will be stored in the profile at the broker. The broker will use this information to maintain the Internet connection to the subscriber. The detected events in the sensing field will be published to the subscriber as soon as possible. If the subscriber is temporarily disconnected, all these events will be cached and delivered to the subscriber when he is connected again.

Similar to the subscribers, each MB has to register to the broker through a "MbRegister" message. Once this message is received, the broker will assign and return a temporary identifier, TID, to the MB. At the same time, a profile "mProfile" will be created in the local memory of the broker for each registered MB. The MBs are usually mobile phones which can relay the subscriptions to the sensors similar to data mules. Hence, each "mProfile" also includes the current location of the MB.

B. Adaptive Location Updates

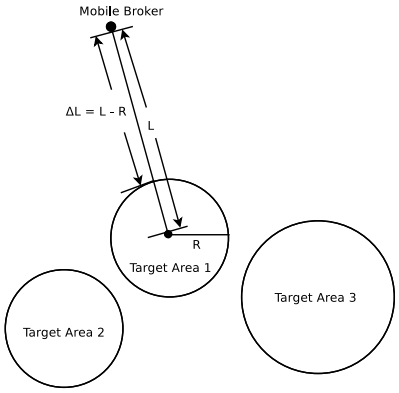

Fig. 5. Situation when MB is outside any target area.

The MBs can relay subscriptions to the remotely deployed wireless sensors when they are within the communication range. Since the broker delivers subscriptions to suitable MBs based on the their locations, the MBs have to update their latest locations to the broker periodically. The time interval for location updates can be configured by a parameter ΔC_{update} in each MB. The location update message "LUpdate" is in a format of $LUpdate[TID, Location]$, where "TID" is the ID of the MB and "Location" includes the latitude and longitude of its current place. The location of a MB is sampled regularly by the GPS in the mobile phone. Since both sampling and location updates to the broker lead to additional computation and communication overhead, we attempt to reduce unnecessary location updates.

As we can see in Figure 5, a MB may be far from any subscribed target area in its walk. During that time, no subscriptions can be relayed to the wireless sensors by the MB. If we measure the distance ΔL between a MB and its closest target area, then we can estimate the best time for the next location update by $\Delta T_{update} = \Delta L / S_{max}$, where S_{max} is the maximum moving speed of the MB. The calculation implies that MB will take at least ΔT_{update} time before it could enter any target area. This mechanism can avoid unnecessary location updates when the MB is far from any target areas.

We divide the moving speeds of the MBs into different ranges. Users who are walking or running usually have their speeds in the range of [0m/s, 8m/s] [18], while their speeds are at most 15m/s when biking [2]. Since the MBs communicate with the SSensors through short-range radio, the MBs that are moving very fast such as driving are not suitable for delivering subscriptions. The broker only selects the MBs from the walking range [0m/s, 8m/s] and the biking range (8m/s, 15m/s]. Whenever ΔT_{update} needs to be updated, the current speed of the MB will be checked against the two ranges. If the current speed belongs to the walking range, S_{max} will be set to 8m/s. Otherwise, S_{max} will be set to 15m/s for calculation.

C. Best-effort Subscription Delivery

Since the subscriptions in our platform specify only the target area, the broker tries its best to deliver subscriptions to all the relevant sensors in that target area. This design supports the joining and removal of stationary and mobile sensors without explicit notifications. A subscription is fully installed when it is delivered to all the sensors of interest in the target area. Consider Figure 6, if all of the SSensors are providing the sensing services requested by the subscriber, the subscription will be delivered to all of the nodes A, B and C.

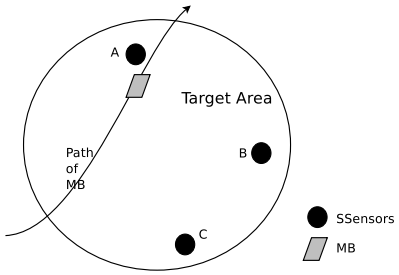

Fig. 6. MB traverses through a subscribed target area.

Figure 7 shows the subscription delivery protocol from a subscriber to the broker. The subscriber first requests sensing data by sending a "SbSUB" message to the broker. The broker then stores the subscription in its memory. In order to distinguish different subscriptions, a unique ID "SpID" will be assigned to each received subscription. Whenever a location update message is received from a MB, the broker will deliver the subscriptions accordingly. Only if the MB is close to the subscribed target areas will receive the subscriptions. In order to maximize the chance of fully installing the subscriptions, the broker will cache all the received subscriptions in its local memory until they are expired.

Best-effort subscription delivery is preferred because the number of MSensors in a target area is higher dynamic. The best-effort approach allows the subscriptions to be installed to as many MSensors as possible. Even though the SSensors

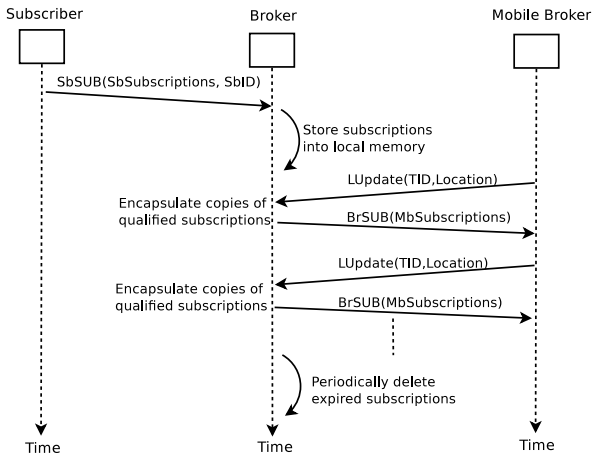

Fig. 7. Subscriptions delivery from subscriber to mobile broker.

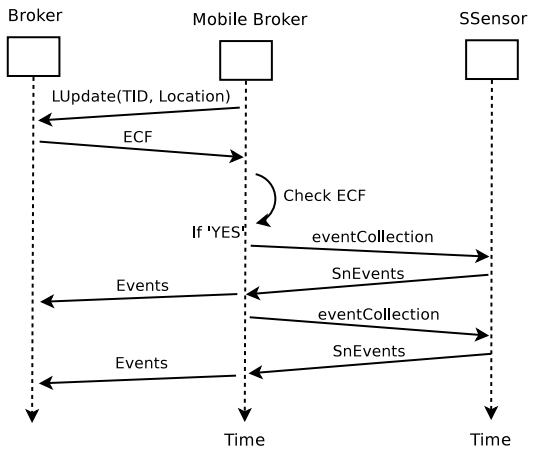

Fig. 8. Event collection from the SSensor.

are not mobile, it is still hard to fully deliver the subscription to them due to their limited communication range. With best-effort subscription delivery, all MBs entering the target area can contribute to relaying the subscriptions to the relevant sensors.

D. Event Data Collection

Once subscriptions are successfully delivered to the sensors, the sensing events are detected and stored. Since SSensors and MSensors have different network connectivities, their event collection mechanisms are different. MSensors such as smart phones are usually equipped with 3G and WiFi connectivities, so that they can report sensing events to the broker on the Internet. Whenever an event is detected by a MSensor, it triggers a "MnEvents" message to its MB module. The MB module then encapsulates the detected event with an "Event" message and sends this message to the broker immediately.

Different from the MSensors, data collection from SSensors adopts a pull-based approach. Since SSensors may not have constant network connectivity, they rely on MBs to collect event data opportunistically. The event data collected by a SSensor have to be cached in the SSensor's local memory before collecting by any MB. A MB broadcasts polling messages "eventCollection" to the SSensors regularly to initiate data collection. Once receiving an "eventCollection" message, the SSensors will deliver all the cached events to the MB with "SnEvents" messages. The MB will then upload the events to the broker as illustrated in Figure 8.

The "eventCollection" messages are unnecessary if no event data has to be collected. Therefore, an event collection flag (ECF) is added to inform the MBs whether they need to perform event collection. When a location update is received from a MB, the broker will check the "SpStatus" of each subscription in the relevant area. If the "SpStatus" is set to "Delivered", it indicates that the subscription is delivered and events have to be collected. Otherwise, a "WaitForDelivery" flag implies that a subscription has never been installed, so no event needs to be collected. The ECF is set to "Yes" or "No" according to the "SpStatus". If the ECF is "Yes", the MB keeps broadcasting "eventCollection" messages. Otherwise, it stops broadcasting.

V. Implementation

We implement our platform on mobile phones and wireless sensors. A broker is implemented on a desktop computer installed with Linux Ubuntu 10.10 and Apache Tomcat 7.0. All programs in the broker are written in Java language. Representational State Transfer (REST) principles are adopted when developing our web services [21]. The broker communicates with other devices in our platform using HTTP messages. In order to manage the subscriptions and the published events, MySQL Database is employed for storage. We present the detailed implementation of the mobile brokers and stationary sensors as follows.

A. Mobile Brokers

We implement the mobile brokers on HTC Hero smart phones with Android OS 1.5. Same as the broker, all programs in the MBs are written in Java language. Each of these smart phones includes a camera, a microphone, an accelerometer and a GPS sensor. Besides, they also support WiFi, Bluetooth, and GPRS/EDGE/3G connectivities. For simplicity, only 3G connection is used in our current implementation. The MBs communicate with the broker using HTTP. They also communicate with the SSensors for opportunistic data collection.

Although many wireless sensors are using IEEE 802.15.4 for communication, most of the existing smart phones could not communicate using this communication standard. We suggest a workaround solution to support communication between mobile phones and wireless sensors as illustrate in Figure 9. In this solution, a sensor node equipped with IEEE 802.15.4 radio is attached to a mobile phone through a USB cable. Since the USB cable needs 5V power supply to enable data transmission, a lithium battery is also connected as external power source. Alternatively, bluetooth sensors could be deployed to support communication with the mobile phones. We also foresee

Fig. 9. Attaching a Tmote Sky sensor to a HTC Hero smart phone.

more lightweight communication protocols become available between mobile phones and wireless sensors, such as the ANT protocol is already available some of the existing smart phones [5].

B. Stationary Sensors

In our implementation, all the SSenors are Tmote Sky nodes running Contiki OS [9]. Both the Contiki OS and our programs for sensors are written in C language. The Rime protocol stack [8] in Contiki OS is used for the network layer communication between the SSensors. The "Discovery" and the "eventCollection" messages are transmitted using the broadcast primitive in Rime, while the "MbSUB" and "InstallStatus" are implemented with the unicast primitive. We implement a serializing mechanism for transmitting sensing data from the SSensors to the MBs. An event packet will be transmitted only when the previous packet is acknowledged. When a SSensor is transmitting event packets to a MB, it will not reply event collection requests from other MBs. From our experiments, the platform server can receive 100% of the transmitted sensing data using the above serializing mechanism.

Since the MBs could only collect data from the SSensors opportunistically, the events detected by the SSensors have to be cached in their local memories. We implement the caching of sensing data with the Coffee file system in Contiki [24]. A memory of two megabytes is allocated for caching sensing data in each SSensor. After reporting the sensing data to the MBs, the SSensor will free its buffer for storing new data.

VI. Experimentation

A. Application Scenario

A hiking trail sensing application is developed based on our ubiquitous publish/subscribe platform. It enables hikers to collect sensing data such as temperature, humidity, light intensity from stationary sensors along the hiking trails as well as to measure their locations and hiking speeds using their mobile phones. With our application, users can subscribe for environmental data (e.g. temperature and humidity) through the Internet before deciding to go hiking or not. They can also plan for their routes in advanced according to the weather and trail condition. Besides, hikers can obtain statistics of their exercises such as their hiking speeds for training purpose. In order to better schedule their activities, they may also want to know the average number of hikers in the trails at different times.

Figure 10 shows the user interfaces of our application on a mobile phone. In this application, the MB module runs on the mobile phone as a background service. Users of this application can gather their locations and speed data while hiking. At the same time, the sensing data collected from the SSensors are also displayed through the user interface.

Fig. 10. The user interface of hiking trail application on a mobile phone.

B. Experimental Settings

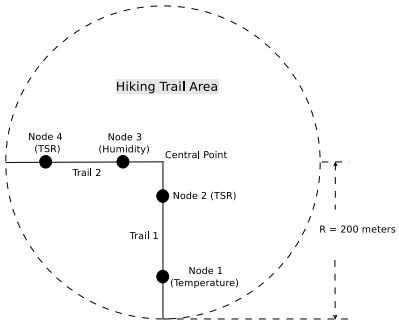

Fig. 11. Experimental settings in a hiking trail area.

We deploy a testbed for our hiking trail application along two walkways outside our laboratory building, namely Trail 1 and Trail 2, as shown in Figure 11. The length of each trail is 200 meters. Four stationary sensors are deployed along the two trails. Node 1 and Node 2 are deployed along Trail 1, while Node 3 and Node 4 are deployed along Trail 2. The intersection point of Trail 1 and Trail 2 is the center of the hiking trail area. The area is a circle with radius of 200 meters. It is also called as the target area in our hiking trail application. Users can subscribe for sensing services from this area by specifying the central location, radius and sensing types. The parameters for experimental settings are listed in Table I.

Fig. 12. Installation of stationary sensors along the hiking trail.

The SSensors are fixed on the poles or trees along the roads as shown in Figure 12. If the SSensors are placed too low, most of the wireless signals will be reflected or absorbed by the ground. The temperature and humidity sensor nodes are placed inside plastic boxes with small holes. The light intensity sensors are put in transparent plastic bags to be exposed in sunlight. The HTC Hero phones are installed with our sensing application, so that they can communicate with the SSensors within the communication range. We installed four different subscriptions in the server at the beginning of the experiment. These subscriptions include the temperature, humidity, light intensity and hiking speed from the target area. We run the experiment from 11 Apr 2011 7pm to 19 Apr 2011 10pm.

TABLE I
EXPERIMENTAL SETTINGS

Parameters	Settings
Length of hiking trail	400m
Number of SSensors	4
Data generation rate at each SSensor	5 samples/hour
Distance between SSensors	100m
Communication power level of SSensors	-5dBm
Number of temperature Sensor	1
Number of light intensity sensor	2
Number of humidity sensor	1
Number of hikers	4
Number of initial subscriptions	4
Experiment period	9 days

C. Sensing Data from Stationary Sensors

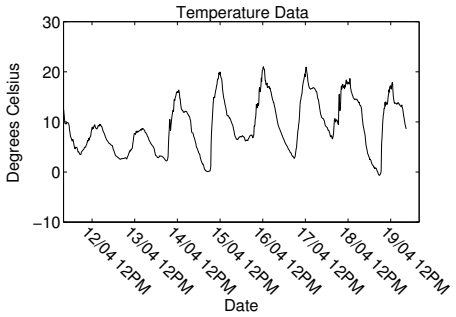

Fig. 13. Temperature data measured by stationary sensor.

The sensing data are collected by the mobile phones from the SSensors and reported to the broker. We plot the sensing data obtained by the broker at the end of our experiment. From Figure 13, the temperature varied from the high of days to the cool of nights in the hiking trail area. In the first two days, the highest temperatures were 9 to 10 celsius degrees, while the lowest temperatures were around 2 to 3 celsius degrees. Then, the temperature increased continuously up to 22 celsius degrees in the following four days. But at the end of experiment, it started to decrease again.

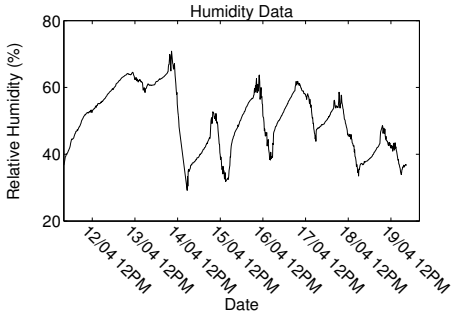

Fig. 14. Humidity data measured by stationary sensor.

From Figure 14, the humidity also fluctuated significantly during the experimental period. In the hiking trail area, the humidity was usually the highest in the early morning and dropped to the minimum in the afternoon. In comparison with Figure 13, the curve of the humidity data was opposite to that of the temperature data. The reason could be the air became drier as the temperature increased.

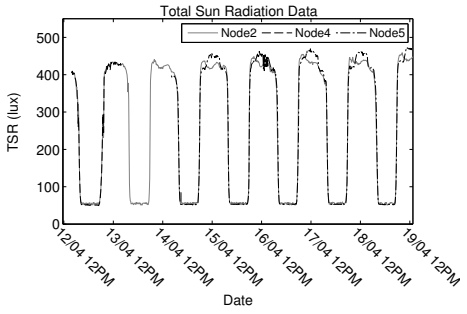

Fig. 15. Light intensity data measured by stationary sensors.

Different from the temperature and the humidity, we believed that the light intensity measured would be affected significantly by the sensor locations. For example, two nodes at different locations may receive the maximum sunlight at different times during the day. We deployed two light intensity sensors, Node 2 and Node 4, along Trail 1 and Trail 2 respectively. Unfortunately, Node 4 was stolen on the third day of our experiment. Then, we had to deploy a new light intensity sensor (Node 5) at a nearby location.

As shown in Figure 15, the light intensity data were collected from three different sensors. Since Node 4 was stolen,

its data only lasted for two days. Node 5 was installed on the third day and it lasted for the following six days. During the daytime, the light intensity measurements from different nodes were not the same. For instance, the maximum light intensity received by Node 5 was higher than Node 2, though Node 2 received its maximum humidity earlier than Node 5. On the contrary, the lowest light intensity from all the three nodes were nearly the same at nights.

D. Sensing Data from Mobile Sensors

Besides the environmental data, the hiking speed of users were collected once they entered the hiking trail area. The mobile sensors, such as GPS on the mobile phones, verified that our platform could support both SSensor and Msensors.

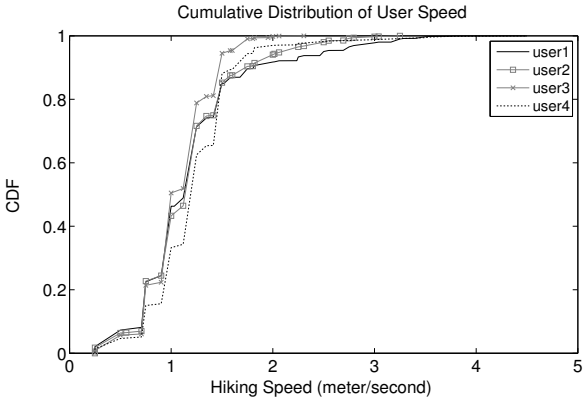

Fig. 16. Distribution of hiking speeds measured by mobile phones.

The speed data collected from the experiment are presented in Figures 16 and 17. In our hiking trail application, the moving speeds of hikers are sampled every two seconds. Figure 16 shows the cumulative distribution of the hiking speeds. We find that over 80% of the collected speeds are below 1.5 m/s. This implies that most hikers are walking most of the time.

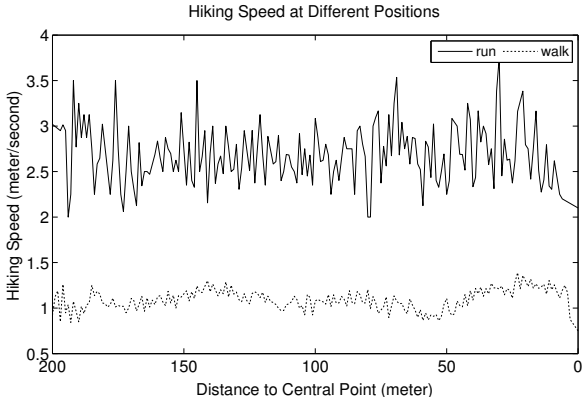

Fig. 17. Moving speed of hikers during walking and running.

The accuracy of the GPS on mobile phone is around 4 to 6 meters. We selected Hiker 1 to study the mobility of users in more details. Since hikers may walk or run during the experiment, we separate the speed data into two categories. Figure 17 shows the running speed and walking speed of Hiker 1 along Trail 1. The x-axis indicates the distance from the entrance of Trail 1 to its exit. The exit of Trail 1 is the center of the hiking trail area. From the figure, Hiker 1 has a walking speed around 0.5-1.5 meters/second and a running speed around 2-4 meters/second. An interesting phenomenon is that the curves of running and walking share similar trend along the x-axis. It implies that road conditions may affect the speed of a hiker. For instance, the speed of hiker may become slower when it is close to the end of the trail.

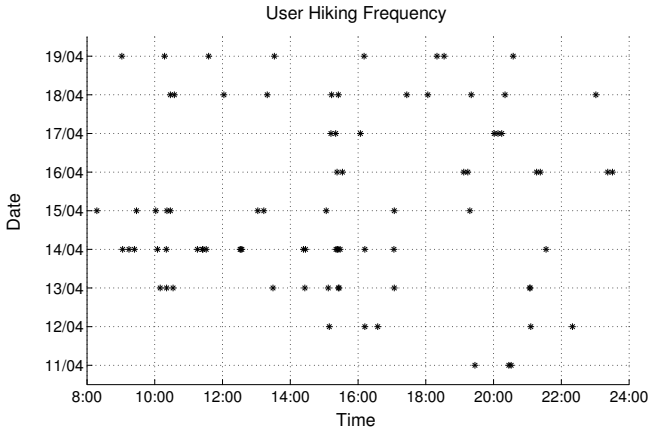

Fig. 18. Visiting frequency of hikers in the sensing area.

We also recorded how many hikers visited the hiking trail area during the experimental period. Figure 18 shows the visiting dates and times of the hikers. Each star marked in the figure represents one visit.

E. Network Performances

We evaluate the data delivery delay, the number of data received, and the communication overhead of our system.

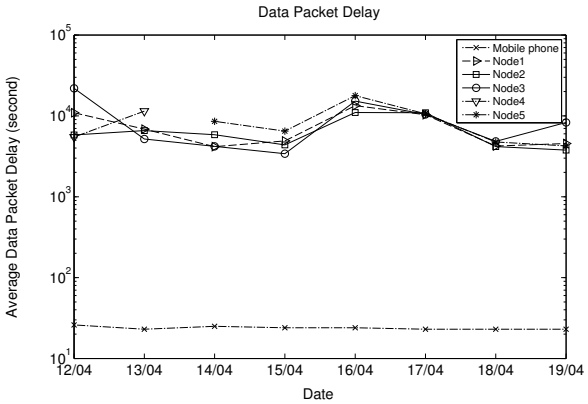

Fig. 19. Data delivery delay.

1) Data Delivery Delay: We divide data delivery delay into two parts. The first part measures the delay from the data

generation time at SSensors to the pick up time by a MB. The second part measures the delay for the MB to upload the data successfully to the broker. We call the first delay as SDelay, and the second delay as MDelay. In cases that the MSensors and the MB are on the same device (e.g. the mobile phone), the data delivery delay only involves the MDelay.

Figure 19 shows the average data delivery delay from the five SSensors and the mobile phones. The mobile phones are utilized as mobile sensors to collect speed data form the hikers. Since the mobile phones have 3G connectivity, the average packet delay of the speed data is only 22 to 23 seconds. However, the data delivery delays from SSensors are much longer. Even the shortest delay from the SSensors is over 3000 seconds, which shows that the SDelay is extremely long for opportunistic data collection.

Since SDelay depends on how often the hikers or MBs enter the vicinity of SSensors, we relate Figure 18 and Figure 19 when analyzing the fluctuation of packet delay. We can see that the delays from SSensors on 12 April and 17 April are obviously higher than the other days (except 16 April). Figure 18 shows that the number of hikes on 12 April and 17 April are obviously less than other days. The results infer that the more frequently the hikers visit the hiking trail area, the shorter the SDelay. Although the numbers of hikes on 13 April, 16 April and 19 April are all equal to 8, the delay on 16 April is much higher than the other two days. From Figure 18, it is easy to observe that the eight hikers on 16 April are clustered into four groups, while hikers on the other two days are more uniformly distributed. This illustrates that not only the number of hikers but also their distribution may affect the SDelay.

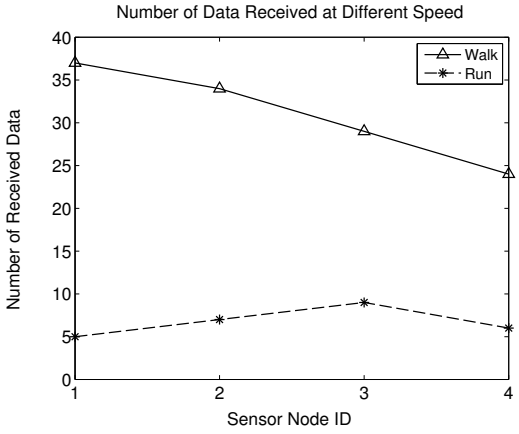

Fig. 20. Number of data received during walking and running.

2) Number of Received Data: Since a MB can only fetch data when it is in the vicinity of a SSensor, its moving speed and the communication range will affect the amount of data received. We study the number of data received when a hiker is running and walking respectively. As shown in Figure 20, mobile broker can receive at most 37 pieces of data during walking, but it can only receive at most 8 packets during running. If the hiker moves slower, it is likely that he can stay longer in the vicinity of a SSensor and receive more data.

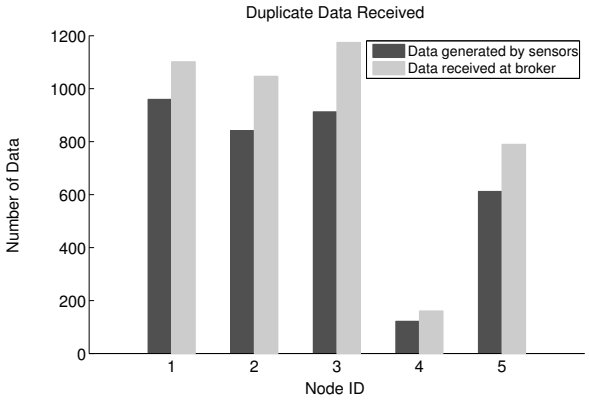

Fig. 21. Duplicate data received from stationary sensors.

Reliable unicast is implemented for data transmission in our system. A data packet is transmitted only when the previous one has been acknowledged. If the current packet is not acknowledged until timeout, it will be restored to the cache file and be retransmitted again. This mechanism ensures that the data packet is either received by a mobile broker or stored at the stationary sensor. As we proved in our experiment, 100% of the generated data are received at broker successfully. However, duplicate packets are also observed as shown in Figure 21. This is because the acknowledgment may be lost even though the mobile broker has already received the data. As a result, duplicate data packets will be transmitted after timeout.

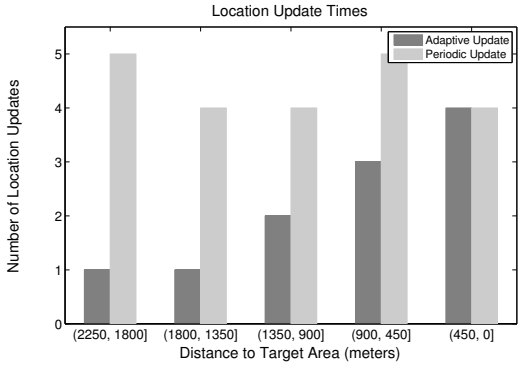

Fig. 22. Number of location updates performed by mobile phones using adaptive update and period update respectively.

3) Communication Overhead: Another improvement in our system is the adaptive algorithm for location updates. The purpose of location update is to check whether the user arrives the sensing field of interest. In the experiment, we compare the adaptive algorithm with the periodic algorithm for location updates. The time interval for location updates is set to 30 seconds in the periodic algorithm. For the adaptive algorithm, the maximum speed of biking is set to 15 m/s. We implement the two algorithms on two different mobile phones. A hiker,

who is carrying these two phones, then start biking towards the hiking trail area from a place 2250 meters away. The location updates are recorded during his ride as shown in Figure 22.

We count the number of location updates when the hiker is moving within different distance ranges. The target area is our hiking trail area. In the adaptive approach, the number of location updates increases when the hiker gets closer to the hiking trail area. In other words, the location of the hiker is reported more frequently to the server when he is approaching the target area. The results show that the adaptive algorithm can reduce the unnecessary location updates effectively compared with the periodic algorithm.

VII. Conclusions and Future Work

In this paper, we proposed and implemented a novel publish/subscribe platform for ubiquitous data access from both wireless sensors and mobile phones. Our platform can support reliable subscriptions and data delivery even for remotely and sparsely deployed wireless sensors. Mobile phones are utilized as mobile mules to relay subscriptions and published data between the Internet users and the remote sensors. Our platform supports data access from both mobile sensors and wireless sensors without any network infrastructure. We implemented our platform on real hardwares and experimented it in a hiking trail application. Experimental results from our outdoor testbed demonstrated successful publish/subscribe services for both stationary and mobile sensors. Network performances such as the packet delay, number of data received and communication overhead have been throughly evaluated.

In the future, we plan to extend our testbed to a larger sensing field that involves more mobile users and wireless sensors. We are also interested in studying the coordination among heterogeneous mobile and sensing devices to further reduce the energy consumption and communication overhead.

Acknowledgments

This work is supported by VINNOVA VINNMER program, VINN Excellence Center for Wireless Sensor Networks (WISENET), SICS Center for Networked Systems (CNS) and ProFun project funded by SSF in Sweden.

References

[1] K. Aberer, M. Hauswirth, and A. Salehi. Infrastructure for data processing in large-scale interconnected sensor networks. In *Proc. of International Conference on Mobile Data Managemen*, pages 198 –205, May 2007.

[2] E.-G. Ahmed, J. K. Kevin, and I. Michael. Predicting bicycle travel speeds along different facilities using GPS data: A proof of concept model. In *Annual Meeting of Transportation Research Board*, 2007.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393 – 422, 2002.

[4] M. Albano and S. Chessa. Publish/subscribe in wireless sensor networks based on data centric storage. In *Proc. of International Workshop on Context-Aware Middleware and Services*, pages 37–42, 2009.

[5] ANT. ANT Android API. http://www.thisisant.com/pages/developer-zone/android-api, 2012.

[6] J. Antollini, M. Antollini, P. E. Guerrero, and M. Cilia. Extending rebeca to support concept-based addressing. In *Proc. of Argentinean Symposium on Information Systems*, Cordoba, Argentina, Sept. 2004.

[7] P. Costa, C. Mascolo, M. Musolesi, and G. Picco. Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 26(5):748 –760, Jun 2008.

[8] A. Dunkels. Rime — a lightweight layered communication stack for sensor networks. In *Proc. of EWSN, Poster/Demo session*, Delft, The Netherlands, Jan. 2007.

[9] A. Dunkels, B. Gronvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. *Proc. of IEEE LCN*, pages 455–462, 2004.

[10] L. Fiege, F. C. Gärtner, O. Kasten, and A. Zeidler. Supporting mobility in content-based publish/subscribe middleware. In *Proc. of ACM/IFIP/USENIX International Conference on Middleware*, pages 103–122, 2003.

[11] D. Frey and G.-C. Roman. Context-aware publish subscribe in mobile ad hoc networks. In *Proc. of International Conference on Coordination Models and Languages*, pages 37–55, 2007.

[12] J.-H. Hauer, V. Handziski, A. Kopke, A. Willig, and A. Wolisz. A component framework for content-based publish/subscribe in sensor networks. In *Wireless Sensor Networks*, volume 4913 of *Lecture Notes in Computer Science*, pages 369–385. Springer Berlin / Heidelberg, 2008.

[13] Y. Huang and H. Garcia-Molina. Publish/subscribe in a mobile environment. *Wirel. Netw.*, 10:643–652, Nov 2004.

[14] U. Hunkeler, H. L. Truong, and A. Stanford-Clark. MQTT-S - a publish/subscribe protocol for wireless sensor networks. In *Proc. of International Conference on Communication Systems Software and Middleware and Workshops*, pages 791 –798, Jan 2008.

[15] A. Kansal, S. Nath, J. Liu, and F. Zhao. Senseweb: An infrastructure for shared sensing. *IEEE MultiMedia*, 14:8–13, Oct 2007.

[16] M. Li and Y. Liu. Underground structure monitoring with wireless sensor networks. In *Proc. of ACM IPSN*, pages 69–78, 2007.

[17] Y. Liu, B.-C. Seet, and A. Al-Anbuky. Virtual brokers for large-scale publish/subscribe in wireless sensor networks. In *Proc. of IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, pages 240–246, 2010.

[18] A. Nummela, T. Kernen, and L. O. Mikkelsson. Factors related to top running speed and economy. *Int J Sports Med*, 28:655–661, 2007.

[19] N. B. Priyantha, A. Kansal, M. Goraczko, and F. Zhao. Tiny web services: design and implementation of interoperable and evolvable sensor networks. In *Proc. of ACM Sensys*, pages 253–266, 2008.

[20] C. G. Rezende, B. P. S. Rocha, and A. A. F. Loureiro. Publish/subscribe architecture for mobile ad hoc networks. In *Proc. of ACM Symposium on Applied Computing*, pages 1913–1917, 2008.

[21] L. Richardson and S. Ruby. *Restful web services*. O'Reilly, first edition, 2007.

[22] K. Shi, Z. Deng, and X. Qin. Tinymq: A content-based publish/subscribe middleware for wireless sensor networks. In *Proc. of International Conference on Sensor Technologies and Applications*, pages 12–17, 2011.

[23] E. Souto, G. Guimares, G. Vasconcelos, M. Vieira, N. Rosa, C. Ferraz, and J. Kelner. Mires: a publish/subscribe middleware for sensor networks. *Personal and Ubiquitous Computing*, 10:37–44, 2006.

[24] N. Tsiftes, A. Dunkels, Z. He, and T. Voigt. Enabling Large-Scale Storage in Sensor Networks with the Coffee File System. In *Proc. of ACM/IEEE IPSN*, San Francisco, USA, Apr. 2009.

[25] A. Wood, G. Virone, T. Doan, Q. Cao, L. Selavo, Y. Wu, L. Fang, Z. He, S. Lin, and J. Stankovic. Alarm-net: Wireless sensor networks for assisted-living and residential monitoring. Technical report, Department of Computer Science, University of Virginia, 2006.

[26] S. Yoo, J. H. Son, and M. H. Kim. A scalable publish/subscribe system for large mobile ad hoc networks. *Journal of Systems and Software*, 82(7):1152 – 1162, 2009.