PMWare: A Middleware for Discovering and Managing Places of Human Interest

Kuldeep Yadav, Abhishek Kumar
Xerox Research Centre
Bangalore, India
kuldeep.r@xerox.com

Prateek Jassal, Vinayak Naik*
EMC Corporation
*IIIT-Delhi
naik@iiitd.ac.in

ABSTRACT

Most of the contextual dependent applications require high level location attributes in terms of places and routes, more than just fine-grained latitude and longitude. Currently, these applications perform place discovery and recognition in an isolated manner with very little coordination and collaboration with each other. This approach is inhibiting for the application developer as well as the end-mobile user, as former has to write redundant and undesirable code for place discovery and the latter's device is subjected to redundant sensing, processing, storage, and higher energy consumption.

In this paper we present a novel framework *PMWare*, which is a middleware that caters to place and route sensing needs of $3rd$ party applications in a unified and integrated manner. It provides an end-to-end service from sensing user's location to discovering high-level location attributes while providing interfaces for managing and storing large-scale human mobility patterns. *PMWare* handles the energy-accuracy tradeoff and uses *triggered-sensing* approach to reduce battery consumption. To demonstrate the end-to-end working of our middleware, we developed an application *PlaceADs* that uses place visiting history of a person to push relevant advertisements. We deployed this application among 16 participants in real-world to check the effectiveness of *PMWare*.

Categories and Subject Descriptors

C.3 [**SPECIAL-PURPOSE AND APPLICATION-BASED SYSTEMS**]: Real-time and embedded systems

Keywords

Context Inference; Place Sensing; Triggered-sensing; Middleware; Contextual Advertisements

1. INTRODUCTION

The Google Play Store has more than 1 million published mobile applications, which were downloaded more than 50 billion times till 2013 [4]. This rise in application downloads are fueled by large growth of smartphones and evolution of mobile network (i.e. $3G$,$4G$). For instance, smartphone users are increasing at an yearly rate of 89% and urban India smartphone users were doubled in numbers in 2013. Various studies have shown that mobile users tend to use only limited number of applications regularly. Therefore, developers are increasingly interested in making their applications smart and proactive by integrating user's contextual information. The presence of multitude of sensors (i.e. GPS, WiFi, Accelerometer) in the mobile phone have made tracking user context easier than before. Gartner predicted that nearly 40% of mobile phones users are going to opt in for mobile sensors based activity tracking by 2015 and context-aware systems will impact nearly 96 billion dollar of consumer spending [2].

Location forms an integral part of a user's context. Mobile phones facilitate ubiquitous location inferencing via interfaces such as GPS, GSM and WiFi [6, 7] that can be directly used by mobile applications. Additionally, other sensors such as accelerometer are used for saving energy in conjunction with these interfaces. However, next-generation mobile applications require high level location information such as places and routes compared to merely geo-coordinates provided by current interfaces. Mobile users spend about $80-90\%$ of their time in different places [5] and therefore, it is now increasingly becoming important for discovering important places in a person's life to provide services.

Currently, there are many applications that are using place-based information including Moves, Google Now, Foursquare, Facebook Places, and Groupon. Also, place-based information can be utilized for setting geo-reminders [8, 9], organizing meetups, participatory sensing [10], content-sharing decisions [11], targeted advertisements[[12], etc. Another section of applications utilize user's route information for different services i.e. pollution exposure estimation [10], healthcare, traffic estimation, ride-sharing, advertisements/recommendations [13].

All of the above applications use their own implementation of place discovery algorithms in an isolated manner with little coordination. One of the downside of this approach is that developers spend lot of time writing redundant code which is not a core-part of their application's functionality and hence, it may be avoided. From a mobile user's point of view also, this approach is ineffective as well as not scalable because it results in redundant sensing, processing, storage, and higher energy consumption. In essence, there are following limitations of current approaches.

1. **Energy-accuracy Tradeoff:** Place discovery algorithms require continuous location sensing. Figure 1 presents the power consumption analysis of different location interfaces in continuous sensing scenarios and clearly, they have very distinct energy profiles. For example, battery duration is almost $11x$ if GSM location is sensed at every minute compared to GPS coordinates. Further, these profiles also change with different location sampling frequencies as shown in Fig-

Middleware'14:Industry short papers, December 08-12 2014, Bordeaux, France.

http://dx.doi.org/10.1145/2676727.2676730.

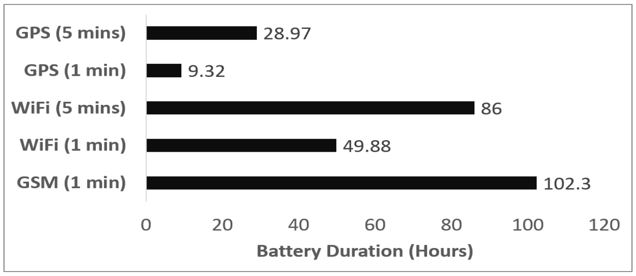


Figure 1: Power consumption analysis of different location interfaces, performed on a HTC $A310E$ Explorer Phone with 1230 mAh battery.

ure 1. Similarly, place accuracy requirements of the mobile applications are different as shown in Figure 2. While, some of the applications (i.e. activity tracking) require room-level accuracy, others such as contextual advertisements could be satisfied with an area level place granularity (i.e. user is in shopping street).

Third-party developers have to deal with energy-accuracy tradeoffs and many times, they end up using simpler interfaces such as WiFi for place-based discovery mechanism even when the application accuracy requirements can be satisfied with location interfaces such as GSM.

2. **Development Complexity:** Third party application developers spend significantly large amount of time in implementing place discovery algorithms and managing person-centric places and routes. It creates an extra burden on them with long development and testing cycles. For example, if a developer wants to create an application that prompts user every morning at home to create a *to-do* list. Even though, inferring user's places (i.e. home) is not a core-part of application functionality but the developer has to write her own functionality in the application to enable this feature.

   Also, a large amount of platform heterogeneity exists for mobile phones i.e. different OSes and their versions. It is very challenging for the developers to implement customizations for all the available platforms.

3. **High Redundancy:** As described earlier, all applications that require places and routes information operate in isolation. Due to lack of coordination between applications, there is redundant and repetitive invocation of location interfaces. Subsequently, place discovery algorithms frequently invoke CPU and use phone memory. All of these operations result in higher battery consumption and poor user experience. Further, some of the place sensing applications (i.e. Moves) prompt users to tag significant places manually. A mobile user has to perform these operations in every application due to lack of coordination.

4. **Lack of customizations:** The accuracy of place and route discovery algorithms depends on many factors such as density of cell towers, coverage of WiFi APs, etc and it may vary across different users and geography. For example, we found that a mobile user is under WiFi coverage for nearly 60% time during a day in India opposed to more than 90% in a developed country such as Switzerland [26]. It is very hard for the developers to consider individualistic customizations while implementing place discovery algorithms.

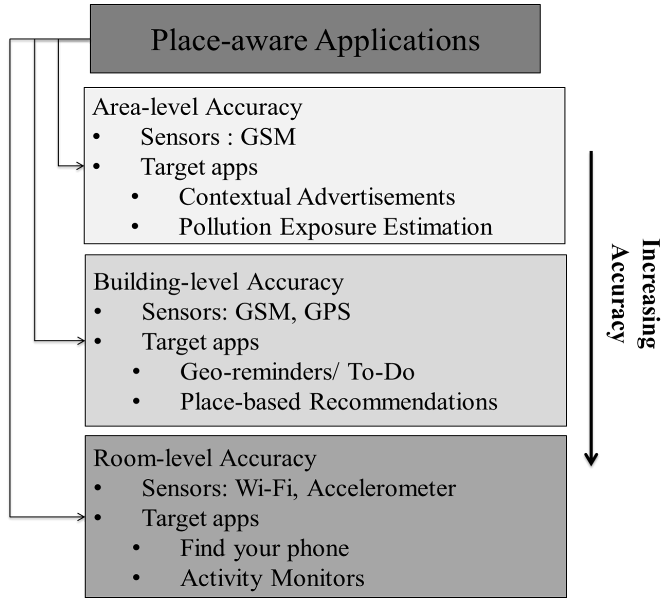


Figure 2: Characterization of place-aware applications

Further, the granularity of a place can be varied from room-level to an area-level based on target application requirements as categorized in Figure 2.

Above challenges motivates the fact that there is a need of a separate framework using which $3rd$ party developers can integrate high-level location attributes (i.e. places, routes) in their applications easily. In this paper, we propose a middleware i.e. *PMWare* that provides an end-to-end service from sensing user's location to discovering high-level location attributes with interfaces for managing and storing large-scale human mobility patterns. In a nutshell, *PMWare* provides a connected application architecture where $3rd$ party applications/services can be plugged with it and subsequently, delegate their place sensing and managing responsibilities to it while focussing on their core services. *PMWare* categorizes the requirements of place-centric applications into three different categories (i.e. area-level, building-level, and room-level) and accordingly, samples location interfaces to minimize overall battery consumption. Specifically, the contributions of this paper are as follows:

1. We develop a novel architecture of a middleware *PMWare*, which is designed to sense low-level location information such as GPS, WiFi, GSM, etc and subsequently, discover and manage high-level location attributes of a user using this data. *PMWare* is based on connected applications architecture where third party application developers can easily integrate with it to delegate their place sensing and management responsibilities.

2. *PMWare* uses *triggered sensing* approach where it continuously samples low energy location interfaces such as GSM continuously and samples high energy location interfaces such as WiFi, GPS based on the demand of connected applications. Using this approach, it strikes right energy-accuracy tradeoff by providing them adequate level of accuracy with minimum possible energy. Additionally, it minimizes redundant sensing, processing, and thus, reducing overall energy consumption on a user's mobile device.

3. As a proof of concept, we develop an application *PlaceAD* that uses place visiting history of a person to push relevant advertisements. We deployed this application in real-world to check the effectiveness of *PMWare* with 16 different participants.

2. SYSTEM DESIGN

PMWare is designed to take place-sensing responsibilities of various applications on a single mobile device in a unified, effective and battery friendly manner. There are many research work which perform place discovery using different location interfaces i.e. GPS [21], WiFi [5], and GSM [26, 17]. *PMWare* builds upon our previous efforts to offer unified APIs for place interface and management [26]. Figure 3 presents the design of end-to-end *PMWare* system as well as different functional components with their corresponding interactions. It also presents a high level overview of how different third party application/services can interact with *PMWare*. In essence, *PMWare* is an amalgamation of two key components i.e. a mobile-based service (PMS) and the *PMWare* Cloud instance (PCI). In this section, we describe all of these components in detail.

2.1 PMWare Mobility Representation

One of the main utility of the *PMWare* is to provide services/modules for managing long-term human mobility patterns. *PMWare* provides easy and consistent representation of these patterns so that it is easy to be consumed by connected applications. In essence, it stores three different high-level location attributes i.e. places, routes, and mobility profile.

2.1.1 Place

A *Place* is defined as a location, where a user stays for significant amount of time, e.g., "Home", "Workplace", "Market". Some of the prior work categorizes a location as a place if user has spent more than 10 minutes at that location [19]. All the place discovery approaches [26] aim to create a unique place signature for every place visited by a user. *PMWare* make use of different location sensing interfaces to discover a place and hence, each place is uniquely identified by a signature which is combination of a set of Cell IDs or a set of WiFi APs or a pair of GPS-coordinates.

P_i = {c_1,c_2,c_3,c_4,c_5} or P_i = {w_1,w_2,w_3,w_4} or P_i = {latitude, longitude}

2.1.2 Route:

The path taken to travel between two places is marked as a route. When different location interfaces are used for route discovery, it comprises of a series of timestamp ordered GPS coordinates or set of time ordered Cell IDs observed during the trajectory.

R_i = {c_1,c_2,c_3.........c_{10}} or
R_i = {g_1,g_2,.........g_{15}} where $\{latitude, longitude\} \in g_j$

2.1.3 Mobility Profile:

Mobility Profile is a spatio-temporal representation of user's mobility which is used to store long-term human mobility patterns. It consists of visited places information along with their respective arrival and departure information, routes information with their start and end time, and social contacts with the encounter start and end time during place visits. In *PMWare*, a day-specific mobility profile is stored and it is represented for any given user X as follows.

$M_X = (P_1, a_1, d_1), (P_2, a_2, d_2), ...(P_n, a_n, d_n) and$
$(R_1, s_1, e_1), (R_2, s_2, e_2), ...(R_m, s_m, e_m) and$
$(H_1, s_1, e_1), (H_2, s_2, e_2), ...(H_k, s_k, e_k)$

2.2 PMWare Mobile Service

It runs on the user's mobile device and primarily responsible for collecting data from different location interfaces, discovering places and routes, and uploading users' mobility patterns on the cloud instance. There is only one instance of PMS running which can be used by multiple connected third party applications, thereby eliminating sensing and processing redundancy. Some of the primary modules of PMS are as follows.

2.2.1 Authentication & User Preferences

Authentication module is responsible for inducing the user device into the *PMWare* system. The device is uniquely identified jointly by its IMEI number and phone email account. It sends a one time registration request to the cloud instance to retrieve an authentication token, which is used for further communication. The authentication token is refreshed periodically based on its expiry time.

User preference module gives control to the mobile users on what kind of place information, it wants to expose to other connected applications. User can configure the place granularity permission for every connected application to preserve her privacy. For instance, a mobile advertisement application want to access place information at building level granularity but user may choose to set permission for only area-level granularity. This module also provides a single control to switch off all place-centric applications whenever she does not like to reveal her place information.

2.2.2 Inference Engine

This module is responsible for data collection from different location interfaces and inferring high level location attributes (i.e. places, routes) from the data. This module provides implementations of place and route discovery algorithms that uses different location interfaces data. An algorithm may use data from one or more location interfaces. Each of these algorithms try to find a unique place signature using which a place can be identified as shown in Section 2.1 and subsequently users's arrival and departure time from the given place is tracked. This module also employs *triggered-sensing* and *opportunistic sensing* approaches to minimize battery consumption. One of such instance is that it tracks GSM-based location information (Cell ID, LAC, MNC and MCC) continuously (i.e. every minute) and duty cycles other sensors such as GPS, WiFi, accelerometer based on the specific application requirements. Sensing GSM-based location information consumes minimal energy because a mobile device is anyway connected to the cellular network. Similarly, scanning WiFi APs is energy-efficient if WiFi is already switched on for data transfers. *PMWare* performs WiFi scans opportunistically to increase accuracy of GSM-only approaches [26].

To bootstrap the system, we implemented following algorithms in *PMWare*. *GCA* is a GSM-based place discovery algorithm that performs clustering on Cell ID data to create place signatures [26]. In this context, clustering is challenging because Cell ID may change even when a user stays at same place due to network load, small time signal fading, and inter-network ($2G$ to $3G$ or vice versa) handoff. Such a change in Cell ID while the user is stationary is called "oscillating effect". GCA models the oscillating effect among Cell IDs using an undirected weighted graph (movement graph) and then performs clustering with the help of heuristics such as edge weights, node degree, etc. *PMWare* uses algorithm described in SenseLoc [5] for place discovery using WiFi data. This algorithm uses tanimoto-coefficient based similarity measure to find unique place signatures as well to detect subsequent arrival and departures from a place. Continuous sensing of WiFi results

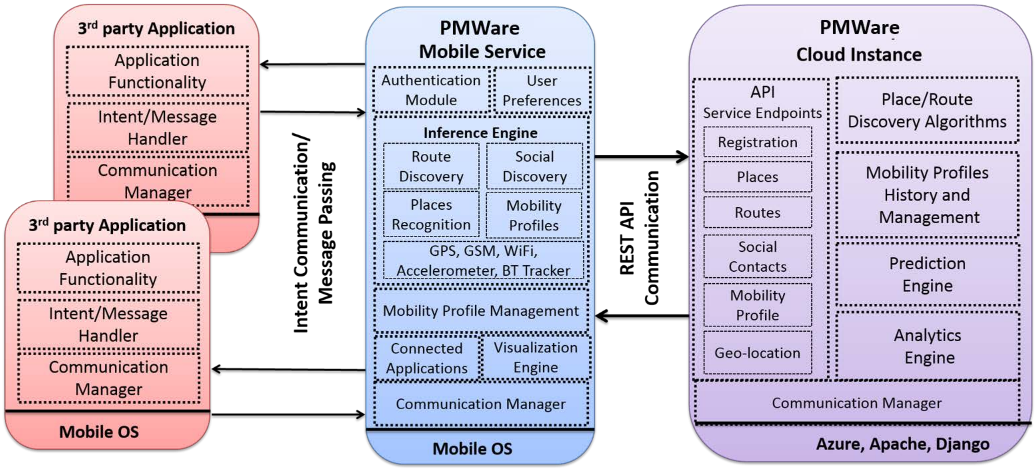


Figure 3: System Design of *PMWare* with its different functional components

in significant battery consumption so accelerometer based activity detector is used to trigger WiFi-based place discovery. In case of GPS, we use Kang et al [16] algorithm for clustering of GPS coordinates according to physical places.

Similar to places, *PMWare* provides a flexibility to specify granularity of route tracking to connected applications. Based on the requirements, *PMWare* has two modes of route tracking, low accuracy mode and high accuracy mode. In low accuracy mode, only GSM-based information is used to track the route information where as in high accuracy mode, WiFi is used to detect place departure and subsequently GPS is used to track the route. Social discovery module is invoked whenever any application requests for information about social contacts that a person encounters on a given place. It detects physical proximity amongst users via their Bluetooth or WiFi data. *PMWare* also allows targeted sensing of social contacts such as monitoring contacts only at the users' workplace.

2.2.3 *Mobility Profile Management Module*

It takes the output of place inference module and subsequently builds mobility profile for a given day because a users' movement pattern is likely to be redundant across days. This module has the responsibility to sync the profile on the cloud instance as well as making this information public to other connected applications based on their requirements.

2.2.4 *Connected Applications Module*

As shown in Figure 3, different third party applications can communicate with *PMWare* using message passing interfaces provided by mobile operating system e.g. intents and broadcasts in Android OS. This module manages all the connected applications and their requirements. For example, a connected application registers an intent to get updates whenever a user visits a new place (event). This module registers the request for the specified event with the *PMWare* system and whenever that event happens, it invokes the appropriate activity of the connected application or makes a broadcast so that other applications can also make use of the information. As described earlier, requirements of the connected applications influence the decision of sensing different location interfaces in *PMWare*. Inference module frequently takes the registered requests and accordingly, invokes appropriate location interfaces.

2.2.5 *Visualization & Communication Management*

The automatic place discovery algorithms may not be completely accurate for some users as well as some of the applications need semantic labelling of the places too (i.e. home, workplace, shopping mall, etc). In such cases, user inputs are necessary to bring confidence in place discovery and labelling process. This module provides visualization of all the places using which user can validate the automatic discovery process and she can also provide a semantic label to the place. This is one of the advantage of *PMWare* where it unifies the human intervention process to make best use of users' attention. Also, human intervention results in enhanced accuracy thus bringing more targeted recommendation/services for the user.

Communication module handles two different kind of communication i.e. REST API based communication with the cloud instance and inter application communication between PMS and connected applications using message passing interfaces.

2.3 PMWare Cloud Instance

It is responsible for storing and managing long-term human mobility patterns, helping mobile service in place/route discovery process, as well as performing advanced analytics and prediction operations to help third party applications and services. In this section, we provide details on some of the key modules of this component.

2.3.1 *Place/Route Discovery Algorithms*

The cloud instance implements some of the place and route discovery algorithms to help *PMWare* mobile service. Some of the place discovery algorithms such as *GCA* [26] are computationally heavy and mobile service offloads this computation to the cloud instance. This is one time computation and after discovery of place signatures, mobile service can track user's visit in those places. Similarly, this module hosts implementation of other miscellaneous algorithms such as route similarity, etc.

2.3.2 *Analytics and Prediction Engine*

Mobility profiles and history module stores the long-term human mobility patterns of a given user. These patterns can be used for predicting user's future mobility and can potentially help third party application/services. As an example, this module can help in answering queries such as below.

1. What is the likely time at which the user typically reaches home in the evening?

2. When will be the next visit of the user for a given place "A"?

3. How frequently user visit shopping malls?

Above queries need significant computation as well as data storage before they can be answered. In such cases, PMS takes help of analytics and prediction engine.

2.3.3 REST APIs

The cloud instance exposes REST based APIs which are used by PMS to invoke cloud-hosted modules. As shown in Figure 3, the API has six different service end points. *Registration* API is used to register a new device to *PMWare* system and assigning an authorization token to the mobile service for accessing other services. *Places* API is combination of different offered services that includes place discovery, retrieval of user visited places, etc. *Route* APIs is also used for discovering routes from raw location data as well as retrieve user-specific routes with optional parameters such as route usage frequency.

Mobility profiles end point hosts combination of services which are responsible for syncing users' profile with the cloud instance as well as to retrieve day-specific mobility profiles requested by connected applications. Social contacts end points helps in storing and retrieving place-specific contacts that are encountered by the user. *PMWare* cloud instance also hosts miscellaneous services such as geo-location API which is used to convert Cell IDs into their approximate geo-coordinates using Open Cell ID and Google Maps geo-location APIs.

2.4 Use Case

We present a step by step use case to demonstrate the interaction among different components of the *PMWare* middleware as well as external components such as connected 3rd party applications. Consider a scenario where a To-Do application intends to alert user with some reminders when the user enters/leaves her workplace. The following steps describe how *PMWare* can be used to meet the requirement of this application.

1. The To-Do application frames its request to get place alerts specifying request parameters like accuracy of place inference and time-period. Assuming that, it specifies that it requires building-level granularity with a tracking between 9 AM to 6 PM. The request also specifies its own intent-filter that will listen to the place alerts broadcasted by *PMWare* mobile service (PMS).

2. The request in the form of an intent is initiated with a target to be captured by the PMS receiver. The receiver captures the request intent, parses the request data to register the place-alert request.

3. The PMS alerts place inference module about the requirements. The inference module accordingly starts sampling the appropriate location interfaces to capture the event. For instance, it may invoke GPS in conjunction with GSM to achieve building-level accuracy.

4. Whenever, the PMS inference engine detects an arrival or departure from the given place, an alert is coupled into an intent and broadcasted targeting the To-Do applications' intent filter.

5. The To-Do application captures the alert intent from PMS with additional place details (i.e. coordinates, name, frequency) and provide reminders to the users accordingly.

3. IMPLEMENTATION DETAILS

We demonstrate the feasibility of *PMWare* by implementing end-to-end system with all the functional components. The *PMWare* cloud instance has been implemented using Django, Python, and Apache web server. The instance runs on a Windows Azure machine and exposes public REST APIs as described in Section 2.3. Due to large scale adoption of Android OS, *PMWare* mobile service is implemented in Android currently but it can be extended to any other mobile platform also. The mobile service can run on any mobile phone which is running Android OS version 2.2 or beyond.

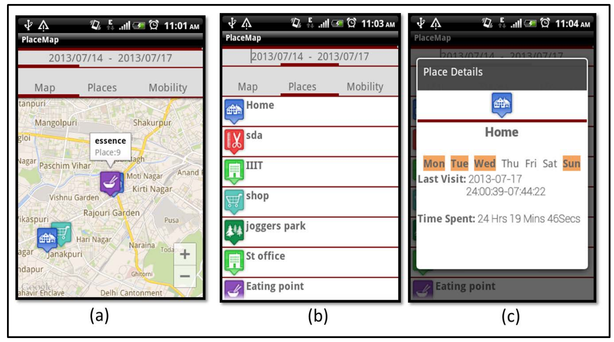


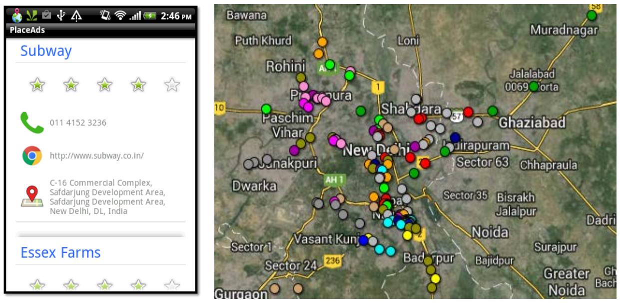
Figure 4: Snapshots of mobility history logging mobile application that uses *PMWare* APIs.

We have packaged *PMWare* mobile service with a life-logging application to create an engaging interface that enables users to validate discovered places as well as to provide a semantic meaning to the places. Figure 4 presents the snapshot of the application which lets user visualize all the visited places on a map interface. All of these places are automatically discovered by *PMWare* algorithms. User have flexibility to tag a place with a customized name and icon, which appears on a list of places as shown in Figure 4.*b*. *PMWare* has a capability to store and manage long-term mobility history of a user. Our mobile application uses that capability to present fine-grained information to the user about her stay time at visited places and visiting days as shown in Figure 4.*c*. A beta version of the application is publicly available on Google Play [3].

Next, we develop a mobile application *PlaceADs* that pushes advertisements and recommendations for new places based on user's mobility profile. *PlaceADs* is developed as a connected mobile application, which uses *PMWare* middleware for sensing and discovering places. For example, whenever a new place is visited, *PlaceADs* gets an intent broadcast from *PMWare* mobile service with the details of the place. *PlaceADs* subsequently fetches targeted contextual advertisements suggesting nearby points of interests such as restaurants, cafes, etc with the corresponding discount offers. As shown in Figure 5a, each advertisement was displayed as a card and the user could like a card by a simple gesture of swiping it on left if it was context relevant and dislike it by swiping it on right if it was not.

4. DEPLOYMENT STUDY

To measure the feasibility and effectiveness of *PMWare* in real-world, we conducted a short deployment study with 16 participants for a given duration of 2 weeks. The only selection criteria used to select the participants was availability of an android phone. Both the mobile applications i.e. *PMWare* and *PlaceADs* were installed in all the participant's phone. Using this deployment study, we wanted to understand the accuracy provided by different *PMWare* algorithms, participant's engagement, and its integra-



(a) (b)

Figure 5: (a) Snapshot of PlaceADs mobile application; (b)Map-based visualization of all the places visited by the participants during user study

tion with third party applications like *PlaceADs*. Participants were provided additional functionality to diary-logging (i.e. logging all place visits manually) to create ground truth. Further, participants provided functionality to like or dislike recommendation generated by *PlaceADs* application.

A total of 123 places were discovered by *PMWare* from the 16 participants as shown in Figure 5b. In total, participants tagged 85 places with their semantic labels i.e. nearly 70% of all the visited places. However, some of the tagged places did not contain departure information. From the rest of the 62 places, we found that *PMWare* using GSM data (augmented with opportunistic WiFi sensing) was able to correctly discover 79.03% of the places, merged 14.52% of places, and divided 6.45% of places. From this user study evaluation, we conclude that *PMWare* has shown effectiveness in discovering places in the wild.

Next, we analyzed the user's input on recommendations provided by *PlaceADs* service. The ratio of total number of likes obtained for the advertisements to the number of dislikes obtained turned out to be $17 : 3$. Further, we observed that most of merged places in *PMWare* were very close to each other i.e. academic building and library which can be easily avoided with the location interfaces such as WiFi. We are planning large-scale deployment going forward with *PMWare* and other associated connected applications.

5. RELATED WORK

With the proliferation of location-based and context-aware applications, there has been a significant amount of research work done to efficiently meet the context and location information needs of such applications. However, most of the prior research work is focussed on sensing location information in an energy-efficient manner and discovering places using one or more location interfaces. None of these work provided an end-to-end middleware solution such as *PMWare*, which can be used by $3rd$ party applications to delegate places and route sensing responsibilities.

Many mobile systems adaptively duty cycle GPS with the help of movement detector and history of movements to save energy. These approaches take help of other sensors from the phones in minimizing GPS usage and provide raw geo-coordinates to applications in an energy-efficient manner [6]. CAPS [23] uses a combination of Cell ID and GPS to build a sequence for user's most travelled paths and later use only Cell ID sequence to provide location coordinates whenever user is at known locations. The ALoc [22] works on the assumption that location accuracy requirements for mobile applications are dynamic in nature and uses multiple mobile phone sensors for localization i.e. inferring lat, lng information. [7] presents a framework to meet the location-sensing requests of mobile applications which conserves energy by substituting GPS with low energy interfaces like GSM, using accelerometer to prevent GPS tracking when user is static. As described earlier, all of these work tend to focus on sensing user's location information (i.e. lat, lng) continuously in an energy-efficient manner and not necessarily focus on place-sensing.

Some approaches use GPS [16], WiFi [5], and GSM data [17] individually or collectively to discover places for a user. These approaches are limited to one kind of place sensing (i.e. room-level or building-level) and does not provide an end-to-end solution such as *PMWare*. Kang et al [16] designed a clustering algorithm to find places using GPS coordinates based on temporal and spatial stay threshold. Senseloc [5] uses repetitive WiFi scans to learn about arrival and departure from a place. To track travel paths, Senseloc uses GPS whenever it detects that user is traveling. Demirbas et al [17] uses GSM data to generate spatio-temporal mobility profile in reality mining dataset [17].

6. CONCLUSION AND FUTURE WORK

Many context-aware applications require high-level location attributes such as places and routes for their functioning. Currently, many of these applications work in isolation and implement their own place and route discovery mechanisms. Due to lack of coordination, user's mobile device is subjected to redundant sensing, storage, and processing, which results in high battery consumption. This approach is cumbersome to the application developer as she has to write separate code for handling place recognition and discovery, which is not the core part of her application.

Our proposed middleware *PMWare* tries to capitalize on the commonality of place information requirement of these applications. Existing $3rd$ party applications can delegate their place sensing and management responsibilities to *PMWare*. It uses triggered sensing approaches to minimize overall battery consumption i.e. it continuously samples low energy location interfaces such as GSM continuously and samples high energy location interfaces such as WiFi, GPS based on the requirement of connected applications to handle energy-accuracy trade-off effectively. We have implemented end-to-end *PMWare* system and shown its effectiveness in real-world using a user-study. We believe that *PMWare* can be used to bootstrap and simplify execution of many third party applications which require to track a user's context/location in a unified, effective, and energy-efficient manner.

As a future work, we plan to open proposed middleware to third party application developers so that they can make use of its capabilities. Also, we intend to extend the capabilities of *PMWare* by integrating other contextual information such as activity tracking as well as integrate techniques to ensure greater privacy and security guarantees for user's sensitive data.

7. ACKNOWLEDGMENTS

This work is partially sponsored by Microsoft Research PhD Fellowship, Nokia Research, and Information Technology Research Academy (ITRA) grants. We would also like to thank all the participants who participated in our deployment study.

8. REFERENCES

[1] LifeMap Google Play application, `https://play.google.com/store/apps/details?id=com.mobed.lifemap`

[2] Gartner Predictions, http://www.gartner.com/newsroom/id/1827614
[3] PlaceMap Google Play application, http://play.google.com/store/apps/details?id=com.iiitd.muc.placemap.
[4] http://goo.gl/Y5qBmq
[5] Kim et al. "SensLoc: sensing everyday places and paths using less energy". ACM SenSys 2010.
[6] Paek, Jeongyeup, Joongheon Kim, and Ramesh Govindan. "Energy-efficient rate-adaptive gps-based positioning for smartphones." Proceedings of the 8th international conference on Mobile systems, applications, and services. ACM, 2010.
[7] Zhuang, Zhenyun, Kyu-Han Kim, and Jatinder Pal Singh. "Improving energy efficiency of location sensing on smartphones." Proceedings of the 8th international conference on Mobile systems, applications, and services. ACM, 2010.
[8] Ludford, P. J., Frankowski, D., Reily, K., Wilms, K., & Terveen, L. (2006, April). Because I carry my cell phone anyway: functional location-based reminder applications. In Proceedings of the SIGCHI conference on Human Factors in computing systems (pp. 889-898). ACM.
[9] Sohn, T., Li, K. A., Lee, G., Smith, I., Scott, J., & Griswold, W. G. (2005). Place-its: A study of location-based reminders on mobile phones (pp. 232-250). Springer Berlin Heidelberg.
[10] Mun et al. "PEIR, the personal environmental impact report, as a platform for participatory sensing systems research." ACM MobiSys 2009.
[11] Talipov E., Chon Y., Cha H., Content Sharing Over Smartphone-based Delay-tolerant Networks, IEEE TMC,2012.
[12] Khan et al. "Mitigating the true cost of advertisement supported free mobile applications". ACM HotMobile 2012.
[13] Kurashima, Takeshi, Tomoharu Iwata, Go Irie, and Ko Fujimura. "Travel route recommendation using geotags in photo sharing sites." In Proceedings of the 19th ACM international conference on Information and knowledge management, pp. 579-588. ACM, 2010.
[14] Vu et al. "Jyotish: Constructive approach for context predictions of people movement from joint Wifi/Bluetooth trace". IEEE PerCom 2011.
[15] Gaonkar S., Li J., Choudhury R.R., Cox L., and Schmidt Al, Micro-Blog: sharing and querying content through mobile phones and social participation,MobiSys ’08.
[16] Kang J., Welbourne W., Stewart B., and Borriello G. "Extracting places from traces of locations." ACM workshop on Wireless mobile applications and services on WLAN hotspots’04.
[17] Bayir M.A., Demirbas M.,PRO, and Eagle N.: Discovering spatiotemporal mobility profiles of cellphone users, WOWMOM 2009.
[18] P. Jassal et al. "Unity: Collaborative Downloading of Content using Co-located Socially Connected Peers". MP2P Workshop, IEEE PerCom 2013
[19] Burbey I.E., Predicting Future Locations and Arrival Times of Individuals. Doctoral Thesis, Blacksburg, Virginia, April 2011.
[20] Do et al. "The Places of Our Lives: Visiting Patterns and Automatic Labeling from Longitudinal Smartphone Data". IEEE Transactions on Mobile Computing, 2013.
[21] Zhou et al. "Discovering personally meaningful places: An interactive clustering approach". ACM Transactions on Information Systems, 2007.
[22] Lin, Kaisen, Aman Kansal, Dimitrios Lymberopoulos, and Feng Zhao. "Energy-accuracy trade-off for continuous mobile device location". In Proceedings of the 8th international conference on Mobile systems, applications, and services, pp. 285-298. ACM, 2010.
[23] Paek, Jeongyeup, Kyu-Han Kim, Jatinder P. Singh, and Ramesh Govindan. "Energy-efficient positioning for smartphones using cell-id sequence matching." In Proceedings of the 9th international conference on Mobile systems, applications, and services, pp. 293-306. ACM, 2011.
[24] Chon, Yohan, Yunjong Kim, and Hojung Cha. "Autonomous place naming system using opportunistic crowdsensing and knowledge from crowdsourcing." In Proceedings of the 12th international conference on Information processing in sensor networks, pp. 19-30. ACM, 2013.
[25] Yadav, Kuldeep, Vinayak Naik, and Amarjeet Singh. "MobiShare: Cloud-enabled Opportunistic Content Sharing among Mobile Peers". IIIT-D Technical Report, IIITD-TR-2012-009, 2012.
[26] Yadav, Kuldeep, Vinayak Naik, Abhishek Kumar, and Prateek Jassal. "PlaceMap: Discovering Human Places of Interest Using Low-Energy Location Interfaces on Mobile Phones", In Proceedings of 5th ACM annual Symposium on Computing for Development (DEV), 2014.