"Follow me": A web-based, location-sharing architecture for large, indoor environments

Polychronis Ypodimatopoulos, Andrew Lippman
The MIT Media Laboratory
20 Ames st.
Cambridge MA, 02139
{ypod,lip}@media.mit.edu

ABSTRACT

We leverage the ubiquity of bluetooth-enabled devices and propose a decentralized, web-based architecture that allows users to share their location by following each other in the style of Twitter[1]. We demonstrate a prototype that operates in a large building which generates a dataset of detected bluetooth devices at a rate of ∼30 new devices per day, including the respective location where they were last detected. Users then query the dataset using their unique bluetooth ID and share their current location with their followers by means of unique URIs that they control. Our separation between producers (the building) and consumers (the users) of bluetooth device location data allows us to create socially-aware applications that respect userÕs privacy while limiting the software necessary to run on mobile devices to just a web browser.

Categories and Subject Descriptors: H.4.0 [Information Systems]: Information Systems Applications; General

General Terms: Experimentation, Security

Keywords: location sharing, bluetooth, agents.

1. INTRODUCTION

Location sharing has been extensively explored in outdoor environments; Web 2.0 services like Google Latitude[2], Foursquare[3] and Brightkite[4] provide location sharing in some social context. These services usually suffer from privacy issues that are occasionally serious[5] and their utility is often questioned due to lack of network effect and fragmentation across different services.

We refocus the benefits of location sharing within indoor environments that are large enough to consider a user's location as a real problem. Examples of such environments include a corporate/university campus, a mall, etc. The more people collaborate within such indoor environments, the more mobile they are and the more useful it becomes to locate an employee or fellow student.

[1]http://twitter.com
[2]http://www.google.com/latitude
[3]http://foursquare.com
[4]http://brightkite.com
[5]http://pleaserobme.com

WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.
ACM 978-1-60558-799-8/10/04.

Location sharing in such environments is often an important problem and its solution does not necessarily need to address network effect and service fragmentation issues. It is relatively easy to inform a building's inhabitants of a service that the building offers, which makes the problem of indoor location sharing even more appealing. We focus our effort in a system with the following characteristics:

simplicity: We need to minimize the hardware and software requirements on the device that will facilitate indoor location inference. Ideally, we would like the user to carry no additional hardware and not have to install any additional software than what an average user may already have.

privacy: Usage of our system should not raise any new privacy issues or require the user to compromise her privacy in any way. By definition, any system that centralizes user location information violates this requirement.

We propose a two-tier architecture that allows users to share their location with each other while addressing the above requirements. First, the building[6] constructs a dataset by collecting the location of devices within it, then, each user queries this dataset in order to infer and subsequently share her location in a decentralized way. While the building maintains a centralized dataset with real-time device location information, it does not need to know which user each device corresponds to. While it is not impossible for the building to infer the relationship between devices and the users that carry them, our system does not introduce any new privacy issues because buildings can already do so in order to track their inhabitants. In such extreme cases, one may also consider the use of one-time device IDs, but this is outside the scope of this paper.

The architecture we propose in this paper involves a device location tracking service that the building offers and a location inference and sharing service that each user controls individually. The latter service is addressable by a unique URl and a congregation of such services, each representing a different user, maintain logical links to other, in the same way that Twitter users follow each other. These services however need not run on the same server, so long as their URI does not change. The distinction between the building's service that produces device location information and the user services that consume that information in order to

[6]From now on we refer to the "building" as the sovereign entity that governs the indoor environment in which users need to share their location.

create and share user location information affords us the following advantages:

- We minimize the complexity of the devices that users need to carry in order to infer their location, by embedding the device location service as part of the services that the building already offers (e.g. WiFi) and by placing the user location inferencing and sharing at a web service that runs outside the user's mobile device and is accessible by software as common as a web browser.
- By storing and managing their inferred location in a service that runs in a trusted environment (e.g. a home or plug[7] computer), users need not have their location information stored in a centralized service.

2. ARCHITECTURE

2.1 Building device scanning service

The building's device scanning service is fairly simple and comprises a set bluetooth scanners connected over ethernet or WiFi to a central server that receives periodic bluetooth device scan results from each scanner. We use Sheeva plug computers[8] as bluetooth scanners because of their diversity and form factor. The server provides a web-based API that allows the following queries against the resulting dataset:

1. Get information for a specific bluetooth device ID (MAC) or name (if detected), such as the scanners that last detected it and corresponding RSSI per scanner.
2. Get list of all scanners currently online, including the textual description of their location and configuration information such as their reporting period.

This API allows the user to easily jumpstart her own location sharing service by discovering her bluetooth ID. It is often hard or even impossible for a user to find the bluetooth ID of her own device, but it is easy to set the name that the bluetooth device uses to advertise itself. By querying the building's service for an arbitrary, yet specific, name that she sets on her device, she can setup her location sharing service with her bluetooth ID.

Scanners poll for new devices in their proximity every 10 seconds and it has been experimentally observed that given a moderate number of bluetooth devices around the scanner (up to about 20), it takes up to 3 consecutive scans for a new device to be detected. As a result, the lag in reflecting a newly appeared device in the dataset is 10-30 seconds.

The API that the building service offers is agnostic of the type of the underlying detection infrastructure; while we are currently using bluetooth, it is straightforward to integrate different types of sensors, such as RFID, to enrich the resulting dataset.

2.2 User location sharing service

Each instance of the location sharing service represents a single user and runs on a unique URI, so two different services may run on different servers. The two main tasks of this service are to:

1. Query the building service using the user's device ID and retrieve its location. It does so on-demand and caches the result for 1 minute.
2. Allow the user to share her location with her "followers" and access the location of users she "follows" herself. The location sharing model is based on Twitter's metaphor of following the tweets of other users, but in our case tweets represent changes in the user's location.

2.2.1 Setup

The user configures the service by providing her bluetooth ID. The building service eases this process by allowing the user to query her bluetooth ID by the device's name, as shown in the previous section. The users exchange URIs representing their individual service the same way they exchange any other URI (the system does not provide an automated way for URI discovery, such as a directory). Then, users exchange "follow" requests and choose whether to auto-accept "follow" requests that they receive. Only the location sharing service of users accepted as "followers" can access the user's location.

2.2.2 Operation

Each instance of the user service exchanges JSON-based updates with its "followers" and offers an HTML-based interface to its owner. By using any bluetooth- and internet-enabled device, the user can easily share and access the location of friends in an indoor environment, by means of a web browser.

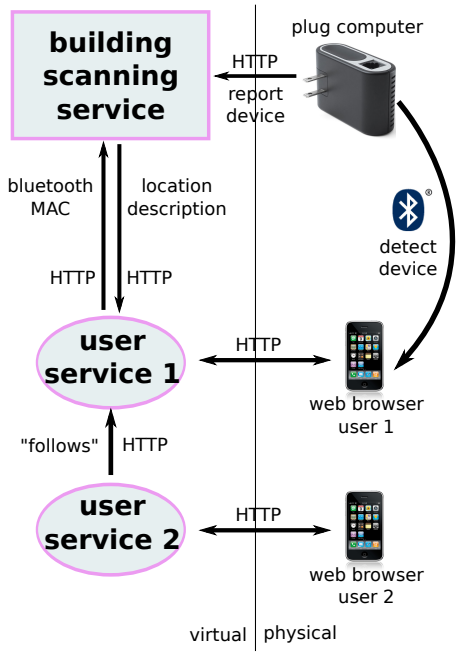

Figure 1: Architectural overview: Building and User location sharing services

Figure 1 shows an architectural overview that involves a building scanning service and a plug computer that reports to it and two user services with the respective bluetooth-enabled devices that represent them. "User service 2" follows

[7]http://www.openplug.org
[8]Sheeva plug computers are kindly donated by Marvell.

the location updates of "user service 1". Figure 2 shows the components of the user location sharing service which is implemented in Python/Django. The core component of the service handles authentication and abstracts access to the DB for the rest of the modules. An extensible number of modules on top of the core handle access to the user's profile, manage "follow" relationships that the user has (friends module) and acquire the user's location from the building service (location module) which gets shared based on the relationships that the friend module stores. The user interface module provides an HTML representation of user locations and allows the user to reinterpret the location description received from the building service, according to user's preference (e.g. translate "Room 515" into "in the office"). A snapshot of the interface is shown in figure 3.

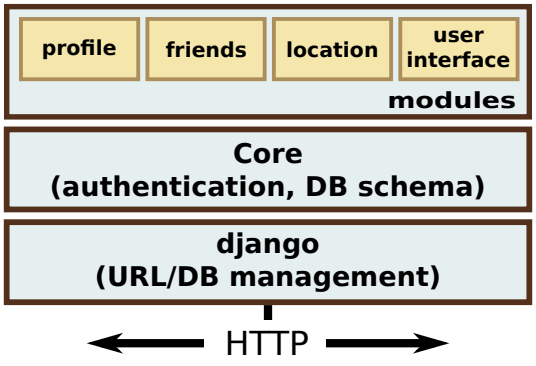

Figure 2: User location sharing service architecture

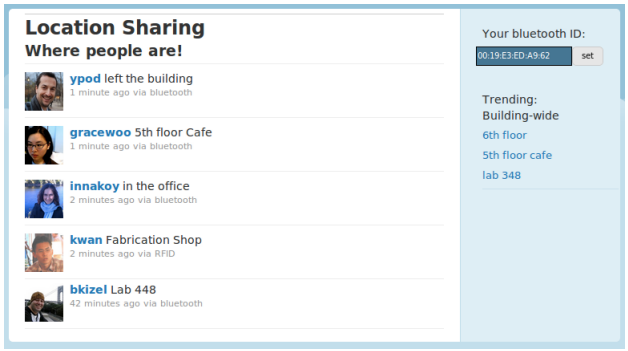

Figure 3: Snapshot of the user location sharing service interface.

3. RELATED WORK AND CONCLUSIONS

The overall cost of the proposed system is kept low, as the only specialized hardware required are the networked bluetooth scanners ($\sim \$50 - \$100$ each). Navigation and location sharing has been explored extensively in the past, both in urban [1, 2, 3, 9, 11, 12] and in a potentially indoor environment [4, 5, 6, 7, 8, 10], but in all cases user location information needs to be stored in a centralized database before it can be shared among users. This limitation raises privacy concerns as users are often reluctant to disclose their location to a third party. Our approach addresses this problem by adopting a decentralized approach in the way users store and share location information.

4. REFERENCES

[1] C. Amick. An architecture for socially mobile collaborative sensing and its implementation. Master's thesis, MIT, 2009. `http://web.media.mit.edu/~camick/camick-thesis-final.pdf`.

[2] M. Bilandzic, M. Foth, and A. De Luca. Cityflocks: designing social navigation for urban mobile information systems. In *DIS '08: Proceedings of the 7th ACM conference on Designing interactive systems*, pages 174–183, New York, NY, USA, 2008. ACM.

[3] S. Casey, S. Lawson, and D. Rowland. Itchyfeet: motivations for urban geospatial tagging. In *NordiCHI '08: Proceedings of the 5th Nordic conference on Human-computer interaction*, pages 435–438, New York, NY, USA, 2008. ACM.

[4] I. Chronis, A. Madan, and A. S. Pentland. Socialcircuits: the art of using mobile phones for modeling personal interactions. In *ICMI-MLMI '09: Proceedings of the ICMI-MLMI '09 Workshop on Multimodal Sensor-Based Systems and Mobile Phones for Social Computing*, pages 1–4, New York, NY, USA, 2009. ACM.

[5] N. Eagle and A. (Sandy) Pentland. Reality mining: sensing complex social systems. *Personal Ubiquitous Comput.*, 10(4):255–268, 2006.

[6] R. José, N. Otero, S. Izadi, and R. Harper. Instant places: Using bluetooth for situated interaction in public displays. *IEEE Pervasive Computing*, 7(4):52–57, 2008.

[7] V. Kostakos. Social networking 2.0. In *CHI '08: CHI '08 extended abstracts on Human factors in computing systems*, pages 3381–3386, New York, NY, USA, 2008. ACM.

[8] H. Mahato, D. Kern, P. Holleis, and A. Schmidt. Implicit personalization of public environments using bluetooth. In *CHI '08: CHI '08 extended abstracts on Human factors in computing systems*, pages 3093–3098, New York, NY, USA, 2008. ACM.

[9] E. Miluzzo, N. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. Eisenman, X. Zheng, and A. Campbell. Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application. In *SenSys '08: Embedded network sensor systems*, pages 337–350, New York, NY, USA, 2008. ACM.

[10] D. Quercia and L. Capra. Friendsensing: recommending friends using mobile phones. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pages 273–276, New York, NY, USA, 2009. ACM.

[11] C. Satchell, M. Foth, G. Hearn, and R. Schroeter. Suburban nostalgia: the community building potential of urban screens. In *OZCHI '08*, pages 243–246, New York, NY, USA, 2008. ACM.

[12] A. Vinciarelli, M. Pantic, H. Bourlard, and A. Pentland. Social signal processing: state-of-the-art and future perspectives of an emerging domain. In *MM '08*, pages 1061–1070, New York, NY, USA, 2008. ACM.