Ubiquitous and Mobile Computing
CS 528: Decentralized Lost and Found with Geolocation

Evin Ugur
Worcester Polytechnic Institute (WPI)
Contents

• Overview
  • Summary/Proposal
  • Vision
  • Related Work
• Methodology/Implementation
  • Application Architecture
  • Location Algorithm
• Results
We all lose stuff!

- People lose items everyday
  - the average person loses $5,591 worth of valuables in a lifetime (Pebblebee)
- Sometimes we find things that people lost
  - There might not always be a lost and found
- These occurrences are very regular on large shared spaces
  - WPI
  - Office
  - Public Event (Festival, Town Meeting)
Proposal

- Decentralized Lost and Found (LAF) app based primarily based on geolocation
  - Camera, and other Ubicomp concepts can be integrated
- App provides way for users to:
  - Report an item of theirs as missing - including the location where it was lost
  - Log unattended items as missing
    - If you lost an item maybe someone already found it
  - Report a missing item as found
    - Notion of Location history - for example if you found some missing headphones in Zoolabs, you can log that and then bring them to the WPI Lost and Found Office
  - Notify users when they are near misplaced stuff (more to come...)
  - Users can interact directly with an item they are reporting or responding to, as well as indirectly with other users using the item as a means of communication.
    - ultimately leading to missing items being resolved
The Vision

- Goal of this project is to show proof-of-concept decentralized LAF service
- Actual Implementations would have to have a layer of social interactions on top of software concept to mitigate against theft.
- Different Social/Usage Models Based on top of Tech:
  - Notion of friends in the app - *When you report to the app that you lost something have option to limit its visibility to only friends*
  - Notion of bounties - have payment interaction around the process
  - Gamification - not mutually exclusive with other approaches
  - Limit Audience - perhaps this service would be very effective exclusively on large communal spaces
    - Large Office Campus
    - TV/Movie Production Sets
Related Work

• Bluetooth Low Energy Attachments
  • Attach a device to your item - items can’t be found retroactively
  • > Chipolo > Pebblebee
  • TrackR uses geolocation when other TrackR users walk near their device - still reliant on hardware and service!

• You can’t install hardware onto everything
• Proposed approach isn’t mutually exclusive
• Hardware trackers are marketed for very important things
  • Wallet - Keys - Dog (attach tracker to collar)
  • Would you use one for your (still important) notepad?
Application Architecture

- Android Application with permission based access to sensors
- Server is hosted by Google App Engine - Part of Cloud Platform
  - Allows easy pulling in of other Google APIs - i.e. PlacePicker
  - Cloud Endpoints provide RESTful interface to backend - Gradle task automatically generates Android client library
- MySQL DB
  - Captures Users, MissingItems, Locations
  - Captures Interactions Between Entities
Finding Nearby Items - Haversine Formula (Theory)

- Great-Circle Distance: *shortest distance between two points on the surface of a sphere (Earth)*
- Haversine formula - computes GCD from latitude and longitude of two points
  - $d =$ distance between points
  - $r =$ sphere radius
  - $\varphi/\lambda =$ lat/lon

\[
\text{hav} \left( \frac{d}{r} \right) = \text{hav}(\varphi_2 - \varphi_1) + \cos(\varphi_1) \cos(\varphi_2) \text{hav}(\lambda_2 - \lambda_1)
\]

\[
\text{hav}(\theta) = \sin^2 \left( \frac{\theta}{2} \right) = \frac{1 - \cos(\theta)}{2}
\]
Finding Nearby Items - Haversine Formula Example SQL

```
SELECT id,
    ( 3959 * Acos(Cos(Radians(37)) * Cos(Radians(lat)) * Cos(
        Radians(lng) - Radians(-122)) +
        Sin(
        Radians(37)) * Sin(Radians(lat)))) AS distance
FROM markers
HAVING distance < 25
ORDER BY distance
LIMIT 0, 20;
```

- Query returns 20 closest location IDs that are within a 25 mile radius of the point lat: -122; lon: 37
- In the LAF Cloud Service, the latitude and longitude are much more precise - and the radius should be very small (Currently experimenting
- 3959 is the radius of the earth (in miles!)
Android Report Found/Lost Flows

Decentralized Lost and Found

Identify a Misplaced Item
Name: Android Ranch Book
Item Description:
has some pencil notes written inside 2nd edition minor wear
Location Description: Fuller zoo labs

No Location Selected
Where were you last? CHOOSE LOCATION
What did you lose?
ID Card
Item Description
Enter any information that might help find identify your missing item
Location Description: Reception Area

Decentralized Lost and Found
Fuller Laboratories
Where were you last? CHOOSE LOCATION
What did you lose?
MacBook Charger
Item Description
has black electrical tape
Location Description: Fuller Lounge

NEARBY PLACES
Worcester Polytechnic Institute
100 Institute Rd, Worcester, MA 01609, United States

Submit Report
Between Now and Tuesday

- Between Now and Code Deliverable Deadline The Largest Hurdle Remaining is integrating the nearby items (which is on the backend) into Android in an intuitive way.
- Other UI tweaks/testing
Thanks!
Questions?
References

- **Tile: Never Lose Anything** [https://www.thetileapp.com/](https://www.thetileapp.com/)
- **Mozy: Lost and Found Reports** [https://mozy.com/about/news/reports/lost-and-found/](https://mozy.com/about/news/reports/lost-and-found/)