Mobile Computing: Prototype Development of a Context Dependent Location Tracking System

Sindre Klavestad
Department of Technology
Kristiania University College, Norway
klavestad@live.no

Tor-Morten Grønli
Department of Technology
Kristiania University College, Norway
tmg@kristiania.no

Tacha Serif
Computer Engineering
Yeditepe University
Istanbul, Turkey
tserif@cse.yeditepe.edu.tr

Gheorghita Ghinea
Brunel University
Dep. Computer Science
Uxbridge, UK State Country
Kristiania University College
Oslo, Norway
George.ghinea@brunel.ac.uk

ABSTRACT

The aim of this paper is to identify whether the development of a context-dependent location tracking prototype would be possible with technology already present in modern smartphones. To do this, a market gap analysis of the current competitors, and how the final product would differentiate from the competitors are given. Furthermore, the chosen technologies and frameworks are described briefly followed by a technical presentation of the prototype and how it works. Finally, the concept and future work are discussed. As a result, the conclusion is that the prototype presented in this paper shows that it is possible to develop a context-dependent location tracking system with technology present in modern smartphones and thus providing proof-of-concept.

CCS CONCEPTS

• ..

KEYWORDS

Mobile Computing · Context-awareness · Location-tracking · Geolocation · Smartphones

1 Introduction

Have you ever started a conversation through text, phone calls or online chat with a friend or family member, wondering where they are? Most likely the answer to this question is yes. As a consequence, there are multiple applications that allow users to share their locations. However, this often involves storage of exact geolocational data on third-party databases, often in different countries than your own, totally out of the user's control. Most of these applications also use a map, displaying the exact locations of friends and family, which is arguably intrusive with regards to privacy.

1.1 Problem Formulation

The aim of this paper is to identify whether the development of a context-dependent location tracking system would be possible with technology already present in modern smartphones. To be more precise, we would like to investigate, whether the development of a mobile application prototype would be able to handle geolocational data, and based upon certain events, export the contextual location of where the user is. As a result, the exact locations of the user will never be used, sent or stored on random third-party databases. The proposed solution will only store the result of the client-side events, triggered by user configured geofences or activities. This data is the result of whether the user is home, at work or out for a walk at this very moment based upon said user configurations. Moreover, because the stored information is displayed in the UI as generalized icons, such as home, work or school, it's contextually different for a family member and a random stranger to know this information. Furthermore, through the development of a reactive solution, taking advantage of background geolocation tracking including mobile and browser clients the positions of friends and family can be monitored at a glance, updated seamlessly from the mobile clients, even when the application is not open.

1.2 Scoping

As a result of the project period being two weeks only, we realized that a fully functioning and deployed mobile/web application including the creation of new users and custom configurations would be difficult. This is why we decided to scope the project down and deliver a minimum valuable product (MVP) for this first iteration. As a result, the product delivered must be considered a prototype where the goal is to provide proof-of-concept and facilitate the potential for including more functionality later on.

2 Related Work

2.1 Background Literature

Location-based Services has come a long way since its first introduction in the early 1990s as part of the Active Badge project (Active Badge, 1992) in the Computer Laboratory, Cambridge University, UK. Location-based services make either use of the fine or coarse location of the user and provide service offerings such as localization and navigation on maps, search, identification, and checking of products (Reichenbacher, 2004).

Notwithstanding its widespread adoption in real-time social systems (Cheng et al, 2011), travel-related applications (Pedrana, 2014), smart cities (Wang, 2016) and its value-adding expedience to commerce (Heinemann and Gaiser, 2015), Location-based services have been a controversial topic from the point of user privacy (Dobson & Fisher, 2003).

Accordingly, Joseph and Choudhury (Joseph & Choudhury, 2009) have proposed an anonymizer system for location data that is called CacheCloak. Their system was a trusted anonymizing server that generated mobility predictions from historical data and submitted the intersecting predicted paths simultaneously to the LBS. The predicted paths were made to intersect with other users' paths so that it ensured no individual user's path could be reliably tracked over time.

Similarly, Gruteser and Grunwald (Gruteser & Grunwald, 2003) proposed middleware architecture and algorithms that can be used by a centralized location broker service. To achieve the required level of anonymity and privacy, their proposed adaptive algorithms adjust the resolution of location information along spatial or temporal dimensions. Using a model based on automotive traffic counts and cartographic material, they try to estimate a realistically expected spatial resolution for different anonymity constraints. Based on their trials, the system's median resolution generated by the algorithms is 125 meters, which is sufficient accuracy needed for E-911 services.

From a different perspective, Li et al. (Li et al., 2017) aim to achieve and enhanced privacy against the insider attack launched by the service providers in mobile online social networks. To tackle this problem, they propose a new architecture with multiple location servers as a secure solution supporting location sharing among friends and strangers in location-based applications. In this system, the user's friend set in each friends' query submitted to the location servers is divided into multiple subsets by the social network server randomly. Each location server can only get a subset of friends, instead of the whole friends' set of the user.

Furthermore, the ability to develop location-based services in the modern web browser using state-of-the-art web technology, has made it easier and more ef- fective to include location-based services across platforms and devices. In the research conducted by Rost et al. (2010), three location-based applications run- ning in the mobile web-browser was explored. This included a self-reporting solution, were users are able to check in at preconfigured location manually, and furthermore two applications taking advantage of the HTML5 geo-location API's.

2.2 Research Gap Analysis

Various location-based tracking systems and services have throughout the decade made an appearance in the market. Either as native applications or as features integrated into already existing applications and solutions. With regards to native applications, most notably is the ”Find My Friends” application developed by Apple (2018), which comes preinstalled on all new iPhones running iOS 9 or newer. Additionally, cross-platform solutions from Life360 (2018), Geozilla (2018) or Familionet (2018) is also worth mentioning. Furthermore, location-based services have also been implemented by various major companies as a feature to their products. Most notably is Google Maps, Facebook Messenger, and Whatsapp where sharing of real-time location can be shared on-demand or Snapchat and its implementation of a "SnapMap", displaying the current location of all your friends. (Kleinman, 2017). Furthermore, if enabled on Android phones, Google will track your movements and display your every move throughout the day on a timeline. (Google, 2018).

So how would a final version of the application proposed in this paper differentiate from the solutions handled by the big companies like Facebook and Apple? There are a couple of reasons why our proposed solution is arguably more privacy-friendly. As we propose client computation of geolocation data, the exact location of the user will never be sent or stored on random third-party databases. The proposed solution will only store the result of client-side events, triggered by user configured geofences or activities. Moreover, because the stored information is displayed in the UI as generalized icons, such as home, work or school, it's contextually different for a family member and a random stranger to know this information. Thus giving the power back to the user with regards to control of their geolocational data, and with whoever they would like to share it.

3 Design and Development

To build a functioning prototype we decided to take advantage of various frameworks and technologies. The sections below are giving a brief description of the frameworks, plugins, and technologies used, and how they were included in the project. Finally, a brief description of the prototype and its functionality is given.

3.1 Meteor Framework

The foundation of the LCTN application is built upon the Meteor framework. "Meteor is a full-stack JavaScript platform for developing modern web and mobile applications. Meteor includes a key set of technologies for building connected- client reactive applications, a build tool, and a curated set of packages from the Node.js and general JavaScript community." (Meteor, 2018) The Meteor framework was chosen to be able to rapidly develop a reactive full-stack prototype, without the need for setup and configuration of a database, server environment, client handling, mobile integration and the connection between said aspects. Based upon the narrow timeframe of the project, this allowed us to utilize more resources towards creating functionality for the actual application.

Installation of Meteor Installation of Meteor was quite simple. By inserting curl https://install.meteor.com/ | sh (Requires OSX/Linux) into the terminal, the latest release of Meteor can be installed.

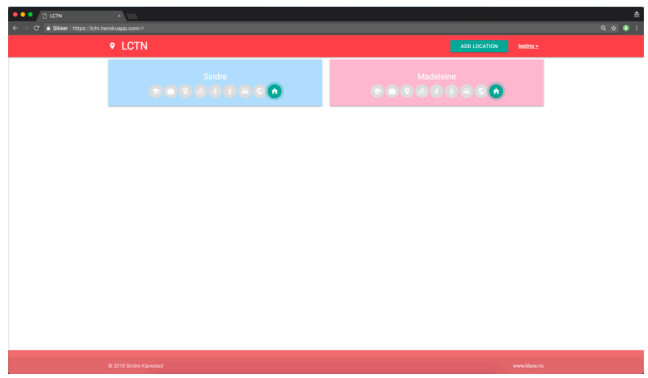

Figure 1 Screen dump of Browser View

3.2 Cordova

Although the native, "out of the box" Meteor framework is meant to build a full-stack web application (as seen in Figure 1), Meteor includes integration of Apache Cordova to be able to include mobile platforms. "Cordova wraps your HTML/JavaScript app into a native container which can access the device functions of several platforms. These functions are exposed via a unified JavaScript API, allowing you to easily write one set of code to target nearly every phone or tablet on the market today and publish to their app stores." (Cordova, 2018) A logged in session on an iOS simulator is shown in Figure 2

Although the solution is only tested for iOS as of writing this, the chosen technologies and frameworks support both iOS and Android platforms. As a consequence, a later implementation of the Android platform should not be a problem.

Integration of mobile platforms to the Meteor project To integrate mobile platforms with the established Meteor project, we simply used the following command in the terminal Meteor add-platform <platform> where platform, in this case, is iOS. When this is done, we are able to run the project locally by using the following command Meteor run

<platform>-device. This requires an installation of tools like XCode and Android Studio, including SDKs and development kits to be able to debug and test in simulators, as seen in emulators and physical devices.

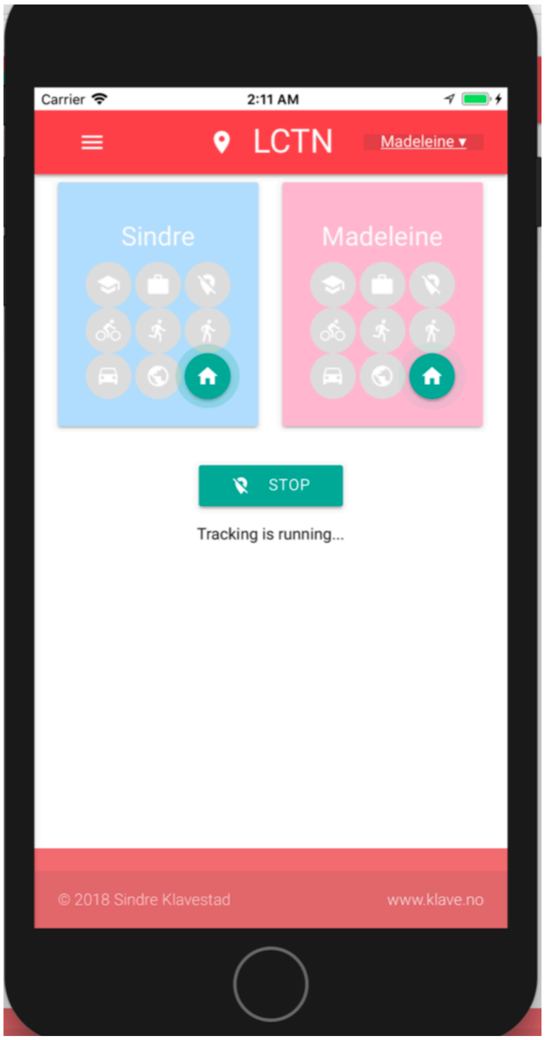

Figure 2 iOS Simulator view

3.3 Cordova Background Geolocation Plugin

To be able to develop a ubiquitous and seamless application we decided to use a technology that enabled tracking of user locations, even when the application is not open. To do this, the Transistorsoftware company has developed a plugin for Cordova which they describe as "The most sophisticated background location-tracking & geofencing module with battery-conscious motion-detection intelligence for iOS and Android." (Transistorsoftware, 2018) The plugin provides methods and functions to gather various geolocation information from the smartphone device. Namely, for this project, the use of activity recognition methods and creation and detection of geofencing events were used.

Furthermore, the plugin provides configurations to enable battery-consciousness intelligent tracking, to avoid draining the battery of the device by tracking 24/7. To avoid this, we have configured the plugin so that the device will have to move a minimum distance of 25 meters before any update events will happen. This means that whenever the device is not moving, the application will not track the user either. Which in turn saves battery power, as seen in Figure 3.

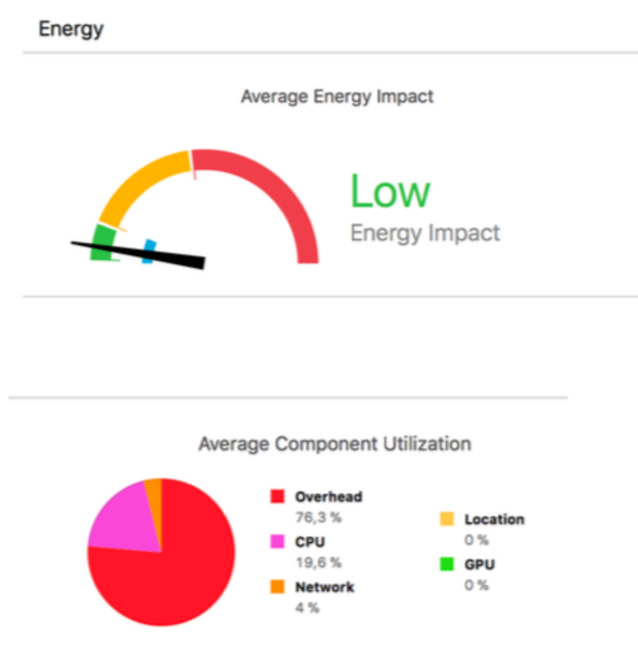

Figure 3 Battery usage in stationary mode

Adding native Cordova plugins to a Meteor project To include Cordova plugins into the meteor project, the following command can be used in the terminal: `meteor add cordova:<plugin name>@<github repository>#version`
where the plugin name, in this case, is cordova-background-geolocation-lt, the GitHub repository is https://github.com/transistorsoft/cordova-background-geolocation- lt.git and the version is 2.6.1.

3.4 MaterializeCSS

With regards to designing an application for multiple platforms and screen sizes, it became clear to prioritize a responsive design. As a consequence, we decided to implement MaterializeCSS into the project. MaterializeCSS is a framework based upon Material Design, a design language ”Created and designed by Google, Material Design is a design language that combines the classic principles of successful design along with innovation and technology.” (MaterializeCSS, 2018a) As a result of this implementation, we had access to the best practice design guidelines through CSS classes. By adding these classes to the HTML elements, we were able to rapidly design a responsive solution. All icons used for the application is also conveniently gathered from the MaterializeCSS library. (MaterializeCSS, 2018b)

Including MaterializeCSS to the Meteor project Because it exists a MaterializeCSS package ready for Meteor, it is simple to include it into the project. The following command fetches and includes the MaterializeCSS package from the meteor packaging environment AtmosphereJS (https://atmospherejs.com/): meteor add materialize:materialize

3.5 Heroku

To be able to fully test the application on the mobile device, running the Meteor solution locally wasn’t sufficient, as the client would not communicate with the servers when it wasn’t on the same network. This made it impossible to check whether the application was able to track if the user is running or walking. This is why we decided to deploy the solution to Heroku. ”Heroku is a cloud platform that lets companies build, deliver, monitor and scale apps — we’re the fastest way to go from idea to URL, bypassing all those infrastructure headaches.” (Heroku, 2018) By doing this, the deployment is available online at https://www.lctn.herokuapp.com for browser devices, however, the native mobile application installed on a mobile device is still needed for fetching the actual location of the users.

3.6 MongoDB

Meteor ships with the popular NoSQL document driven database MongoDB natively, and is used to store current location of users, and login information. However, when deploying the solution to Heroku, we had to implement mLab to the Heroku platform. ”mLab is a fully managed cloud database service featuring automated provisioning and scaling of MongoDB databases, backup and recovery, 24/7 monitoring and alerting, web-based management tools, and expert support.” (mlab, 2018).

Through the free sandbox version, we were now able to deploy a fully working prototype to using a cloud database to insert and fetch data. To be able to do test runs on the mobile device connecting to the cloud Heroku + mLab solutions, however, we need to include some parameters in the terminal start command. The following command will connect to Heroku and mLab cloud solutions: env ROOT URL=<Root URL> MONGO URL=<MONGO URL> meteor run <platform> -device --mobile-server <Root url Where the Root URL is https://lctn. herokuapp.com and MONGO URL is mongodb://heroku z399m1w3:vsbk731hg 5eimd4kqb5pf5t948@ds229918.mlab.com:29918/heroku z399m1w3

When updating the locations of a user, the database location collection is built with the following structure, where each of the 9 locations has an individual document like this one:

```
{
”location”: ”<locations >”, ”isSindreHere”: false ,
”isMadeleineHere”: false
}
```

Furthermore, the rest of the application will reactively update whenever one of the attribute is updated in the database. Meaning that whenever the ”isSindreHere” attribute turns to true instead of false, the location in the UI will be updated accordingly.

4. Results & Discussion

The prototype includes a functional location tracking system which can be accessed through web and mobile view, as seen in Figure 1 and Figure 2. Although still just a prototype, the following list displays the functionality included in this prototype as of this iteration:

- – Full-stack environment deployed to cloud solution (https://www.lctn.heroku app.com)
- – Login/Authentication using username/password.
- – Login/Authentication using Github.
- – Reactive UI that updates whenever the database updates.
- – If logged in as Sindre or Madeleine users on a mobile device running the

cordova application, the user is able to start and stop geolocation tracking, which will update the respective user location whenever a geofence or activity event is triggered.

- – The application will be based upon motion and geofence functionality in order to update one of nine generalizing icons in the UI. This includes Home, School, Work, Running, Walking, Bicycling, Driving, No location and "public"(standing still somewhere that is not defined).

"We believe this is only the beginning of the explosion of mobile services and applications; an even bigger boom will occur when many of these apps will be written for the mobile web instead" (Rost et al., 2010).

6. Conclusion and Future Work

The aim of this paper was to identify whether the development of a context- dependent location tracking system would be possible with technology already present in modern smartphones. To do this, a review of related work, and how the final product would differentiate from the state-of-the-art solutions was given. Furthermore, the chosen technologies and frameworks are described briefly followed by a technical presentation of the prototype and how it works.

The prototype so far includes a full-stack reactive client/server environment which will be updated by the contextual location of Sindre or Madeleine users. This information is based upon the current locations gathered from mobile application clients where the Sindre and Madeleine users are logged in. Now, this concludes that the core functionality for tracking of users through modern smart- phones work. However, the application would still need a couple more development iterations in order to be commercialized and sold to the market. As an example, there is a need to build logic for user authentication to access the location of other users. This could be done by creating a group system, where users can add their friends and/or family to multiple groups, and all users in a group would be displayed similarly to how the Sindre and Madeleine users are displayed in the current version. Furthermore, as of the prototype version, the geofences is hardcoded into the client-side logic. In a production version, the user would have to configure their own geofences after the application is installed, and moreover having access to a configuration screen where the personal information can be added/edited. And as a result, to avoid storing this information including the exact location (which a geofence object includes) on third-party databases, we propose implementing the built-in localStorage functionality from Cordova, to store sensitive user data directly on the device. By doing this, the users would be able to include personalized geofences, without the need to worry about exposure of sensitive data to third-parties. Moreover, the ability to customize geofences will allow the users to include the geolocations that will contextually fit their unique lives, and not just Home, School or Work. Furthermore, the application could even come preinstalled with various public places like parks, landmarks or airports for the users to be able to easily include as their own geofences.

However, although the product will need a few more development iterations to be commercialised and sold to customers, we would like to conclude that the MVP / prototype presented in this paper shows that it is possible to develop a context-dependent location tracking system with technology present in modern smartphones, thus providing proof-of-concept.

7 References

Apple (2018). Find my friends. https://itunes.apple.com/us/app/find-my-friends/id466122094?mt=8. Online; accessed 16 February 2018.

Cordova (2018). Supported platforms. https://cordova.apache.org/. Online; accessed 14 February 2018.

J. E. Dobson and P. F. Fisher, "Geoslavery," *IEEE Technology and Society Magazine*, vol. 22, no. 1, pp. 47–52, 2003.

Familionet (2018). Familio. https://www.familo.net/en/. Online; accessed 16 February 2018.

Geozilla (2018). Geozilla. https://geozilla.com. Online; accessed 16 February 2018.

Google (2018). Timeline. https://www.google.com/maps/timeline?pb. Online; accessed 16 February 2018.

Gruteser, Marco, and Dirk Grunwald. "Anonymous usage of location-based services through spatial and temporal cloaking." Proceedings of the 1st international conference on Mobile systems, applications and services. ACM, 2003.

G. Heinemann and C. Gaiser, "Study: Status and potential of location-based services," in Social-Local-Mobile. Springer, 2015, pp. 155–185.

Heroku (2018). What is heroku? https://www.heroku.com/what. Online; accessed 15 February 2018.

Kleinman, J. (2017). How to share your location in whatsapp and other apps. https://lifehacker.com/ whatsapp-just-added-live-location-sharing-heres-how-to-1819658511. Online; accessed 16 February 2018.

Li, J., Yan, H., Liu, Z., Chen, X., Huang, X., & Wong, D. S. (2017). Location-sharing systems with enhanced privacy in mobile online social networks. *IEEE Systems Journal*, *11*(2), 439-448.

Life360 (2018). Life360. https://www.life360.com/. Online; accessed 16 February 2018.

M. Pedrana, "Location-based services and tourism: possible implications for destination," Current issues in Tourism, vol. 17, no. 9, pp. 753–762, 2014.

MaterializeCSS (2018a). About. http://materializecss.com/about.html. Online; accessed 12 February 2018.

MaterializeCSS (2018b). Icons. http://materializecss.com/icons.html. On- line; accessed 11 February 2018.

Meteor (2018). What is meteor? https://guide.meteor.com/. Online; accessed 14 February 2018.

Meyerowitz, Joseph, and Romit Roy Choudhury. "Hiding stars with fireworks: location privacy through camouflage." Proceedings of the 15th annual international conference on Mobile computing and networking. ACM, 2009.

mlab (2018). mlab: Database-as-a-service for mongodb. https://mlab.com/ company/. Online; accessed 15 February 2018.

Rost, M., H. Cramer, N. Belloni, and L. E. Holmquist (2010). Geolocation in the mobile web browser. In Proceedings of the 12th ACM International Conference Adjunct Papers on Ubiquitous Computing - Adjunct, UbiComp '10 Adjunct, New York, NY, USA, pp. 423–424. ACM.

Transistorsoftware (2018). Cordova background geolocation. https://github. com/transistorsoft/cordova-background-geolocation-lt. Online; ac- cessed 15 February 2018.

C. Wang, "Location based services and location based behavior in a smart city," Ph.D. dissertation, Universite ́ de Lyon, 2016.

Z. Cheng, J. Caverlee, K. Lee, and D. Z. Sui, "Exploring millions of footprints in location sharing services." ICWSM, vol. 2011, pp. 81–88, 2011.

(Reichenbacher, 2004) T. Reichenbacher, Mobile cartography: adaptive visualisation of geographic information on mobile devices. Verlag Dr. Hut Munchen, 2004.

(Active Badge, 1992) https://www.cl.cam.ac.uk/research/dtg/attarchive/ab.html