Better Approach to Geofence Detection

Andrea Sghedoni MATR.0000736038*,
* DISI, University of Bologna, Italy

Email: andrea.sghedoni4@studio.unibo.it

***Abstract*—My Location Alert is a smartphone application developed for the Android operating system. The app allows you to create fences, selecting a center and a range, and capture all ENTER/EXIT events. Whenever you experience any of these events, it is automatically sent an alert SMS to a mobile (previously selected). The main goal of the project, however, is to compare the battery consumption of two strategies/services for location monitoring in time. The first strategy is based on the simple polling strategy, while the second adopt a smarter auto-adaptive approach to avoid an excessive battery consumption, which would make the app unusable.**

I. Introduction

The geofencing approach means taking monitored geographical area and giving awareness to the device of the EXIT/ENTER/REMAIN events in one place. Location Services is one of the most important cause of battery discharge in smartphones. If these services must to be continuous in time (with active jobs in background), it is likely the user to view a major deterioration of the battery. In recent years, Google is committed to the improvement of these activities, especially to reach an optimum point between precision and consumption. Please note that the location can be basically made from two mechanisms: the first concerns the use of the GPS sensor and has an important consumption of battery in the phone, while the second concerns the use of the network, even if in this precision is less, in comparison to the use of GPS sensor. Most of geofence mechanisms adopt a polling strategy. which controls the position at constant intervals (5 sec, 10 sec, ...). This method, if it is merged with the use of GPS, drains battery in few hours. Considering also that the user, of course, does not use the smartphone only for geofencing, but we must also take account of all the other apps that users install, use and need. The main goal project is to implement an intelligent mechanism for which the position is monitored frequently and with the GPS only if you are close to a fence, while controlling the position less frequently (and via network, to a lower consumption) as that the device moves away from the user-selected geofences. The approach relies on an auto-adaptive mechanism, where the algorithm tries to use certain resources only when necessary, details will still in Section III. The project was developed as part of Mobile Systems exam at Alma mater Studiorum, Bologna. The main use case of My Location Alert app is automatically alert a phone number (via SMS) of entry/exit events from a specific geographic region.

II. Related works

Fornisci una breve rassegna di articoli di ricerca, software, prototipi o tecnologie che sono collegate in qualche modo al problema affrontato nel progetto. Tutti i lavori devono essere referenziati ed inseriti nella Bibliografia.

III. Architecture

Basically, the app is composed with the two traditional components, such as frontend and backend. The first component is represented by the graphic interface with which the user interacts, such as maps, registration forms fence etc. The most interesting part is the back-end, which is composed of: Entity, SQLiteManager, Controller, Services.

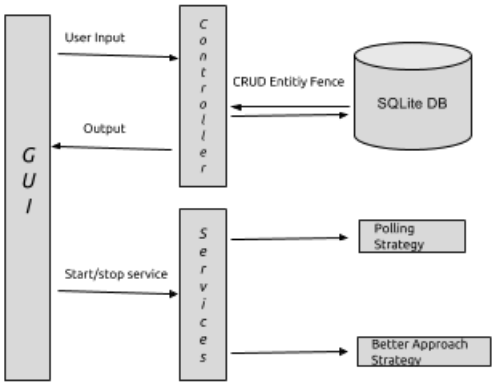

Fig. 1. Architecture

Entity map a fence in a java object that contains all the information needed to manage it, such as to address, latitude, longitude, range, state, activation, event to detect.
SQLiteManager manages the database, provides methods for CRUD(create, read, update, delete) data of the fence. This allows you to record your preferences. The DB is composed of a single table, in which each entry corresponds to a fence, registered by users and each column map an attribute of the Java class Fence.
The *Controller* is an object that implements Singleton pattern, only one instance of that class is created and accessed. It exposes data and methods that should be available and accessed from anywhere in the project. Mainly, it provides the current user fence list and acts as interface to the SQLite

DB, providing functions that implement CRUD verbs.
Services implement the monitoring and detection strategies of registred fences, which run in the background even when the app is not open. Only one of the two services can be active at a given time. The performance differences between the two services implemented is the true heart of the project, you will see in the results that the polling startegy produces a battery consumption exaggerated relative to the other auto-adaptive strategy.

IV. Implementation

[Descrivi come eśtato implementato il sistema, ossia tecnologie utilizzate, linguaggi, APIs, etc. Nel caso, fornisci pseudo-codice degli algoritmi piuínteressanti sviluppati nel progetto.] Android Studio is the official IDE for Android, where you can test code on different device emulators and monitor log. The development of a native Android app requires the use of object-oriented language JAVA. With regard to the persistence of the information (basically the Fence objects with their attributes) entered by the user it has been necessary to create and maintain a DataBase SQLite. We have used various services of google map api Android, from the representation of the map (markers, fence circles) to use of location services (location update requests). In addition, the Geocoding service allows the user to select the center of fence, with an address directly and not with latitude and longitude. Git is used as a version control system, very important for step-by-step development and progress monitoring.

Distance(meters)	f(distance)
< 8000	5
8000-30000	4
30000-50000	3
50000-100000	2
100000-200000	1
> 200000	0

TABLE I
Distance evaluation

Speed(Km/h)	f(speed)
> 130	5
130-100	4
100-60	3
60-20	2
< 20	1

TABLE II
Speed evaluation

Direction	f(direction)
Yes	5
No	0

TABLE III
Direction evaluation

$$\alpha = 0.8 \tag{1a}$$
$$\beta = 0.1 \tag{1b}$$
$$\gamma = 0.1 \tag{1c}$$

$$\boxed{f_i = \alpha f(distance) + \beta f(direction) + \gamma f(speed)} \tag{2}$$

f_i	Interval(second)	LocationRequest Precision
5	5	$PRIORITY\_HIGH\_ACCURACY$
4	30	$PRIORITY\_BALANCED\_POWER\_ACCURACY$
3	60	$PRIORITY\_BALANCED\_POWER\_ACCURACY$
2	180	$PRIORITY\_LOW\_POWER$
1	300	$PRIORITY\_LOW\_POWER$
0	480	$PRIORITY\_LOW\_POWER$

TABLE IV
Evaluation strategies

V. Performance evaluation

Illustra qui i risultati sperimentali (simulazioni o esperimenti) che catturano le prestazioni del sistema realizzato. Chiarisci quali sono gli indici di stima e come sono calcolati. Inserisci un breve commento per ogni grafico.

VI. Conclusioni

Conclusioni, possibili sviluppi futuri e limitazioni del progetto realizzato

References

[1] Lista Autori Titolo Lavoro *Nome Rivista o Convegno*, pagine, anno pubblicazione.