Masters Program in Geospatial Technologies

INFORMATION PROVISION IMPROVEMENT WITH A GEOFENCING EVENT-BASED SYSTEM

Aida Monfort Muriach

Dissertation submitted in partial fulfilment of the requirements
for the Degree of *Master of Science in Geospatial Technologies*

NOVA IMS Information Management School
UNIVERSITAT JAUME·I
WESTFÄLISCHE WILHELMS-UNIVERSITÄT MÜNSTER

INFORMATION PROVISION IMPROVEMENT WITH A GEOFENCING EVENT-BASED SYSTEM

Dissertation supervised by

Prof. Dr. Roberto Henriques
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa
Lisbon – Portugal

Co-Supervised by

Prof. Dr. Francisco Ramos Romero
Dept. Lenguajes y Sistemas Informaticos
Universitat Jaume I
Castelló – Spain

Prof. Dr. Christian Kray
Institute for Geoinformatics
Westfälische Wilhelms-Universität Münster
Münster – Germany

February 2015

ACKNOWLEDGEMENTS

I would like to thank my supervisors Francisco Ramos, Roberto Henriques and Christian Kray for their comments and guidance during the course of this dissertation. Special gratitude to Francisco Ramos for his continued support.

Thanks also to all of our professors, to the directors of the programme, to the erasmus coordinators, to Joan Avariento for his technical help with ArcGIS and to Dori Apanewicz for solving our doubts.

I will never forget all my classmates and all the moments that made my experience richer and happier.

Finally, I would like to thank all my family, specially Saray, Tica and Edu and all my friends for their advice.

INFORMATION PROVISION IMPROVEMENT WITH A GEOFENCING EVENT-BASED SYSTEM

ABSTRACT

Nowadays there is a big percentage of the population, specially young users, which are smartphone users and there is a lot of information to be provided within the applications, information provision should be done carefully and should be accurate, otherwise an overload of information will be produced, and the user will discard the app which is providing the information.

Mobile devices are becoming smarter and provide many ways to filter information. However, there are alternatives to improve information provision from the side of the application. Some examples are, taking into account the local time, considering the battery level before doing an action and checking the user location to send personalized information attached to that location.

SmartCampus and SmartCities are becoming a reality and they have more and more data integrated every day. With all this amount of data it is crucial to decide when and where is the user going to receive a notification with new information.

Geofencing is a technique which allows applications to deliver information in a more useful way, in the right time and in the right place. It consists of geofences, physical regions delimited by boundaries, and devices that are eligible to receive the information assigned to the geofence. When devices cross one of these geofences an alert is pushed to the mobile device with the information.

KEYWORDS

LBS
Geofencing
Geotrigger
Trigger
Mobile Application
Push Notification
GIS
Geoprocessing
Polygons
ArcGIS
GeoJSON
Xcode
SmartCampus
SmartCity

ACRONYMS

LBS – Location Based Services

GPS – Global Positioning System

AGPS – Assisted Global Positioning System

LBA – Location Based Applications

IDW – Inverse Distance Weighted

JSON – JavaScript Object Notation

IDE – Integrated Development Environment

SDK – Software Development Kit

GPX – GPS eXchange

API – Application Programming Interface

APNs – Apple Push Notification service

cURL – c Uniform Resource Locator

INDEX OF THE TEXT

INDEX OF TABLES

INDEX OF FIGURES

1.- Introduction

1.1.- Overview

The current expansion of smartphones will reach by the end of 2014, 1.76 billion users in the world[1], 25% more over the users in previous year 2013. Currently, there are 19 countries with more than 50% smartphone penetration[2], which means that more than the half of the population of these countries has a smartphone, these including South Korea, Australia, Japan, Norway, Sweden, Denmark, United Kingdom, Netherlands or United States of America.

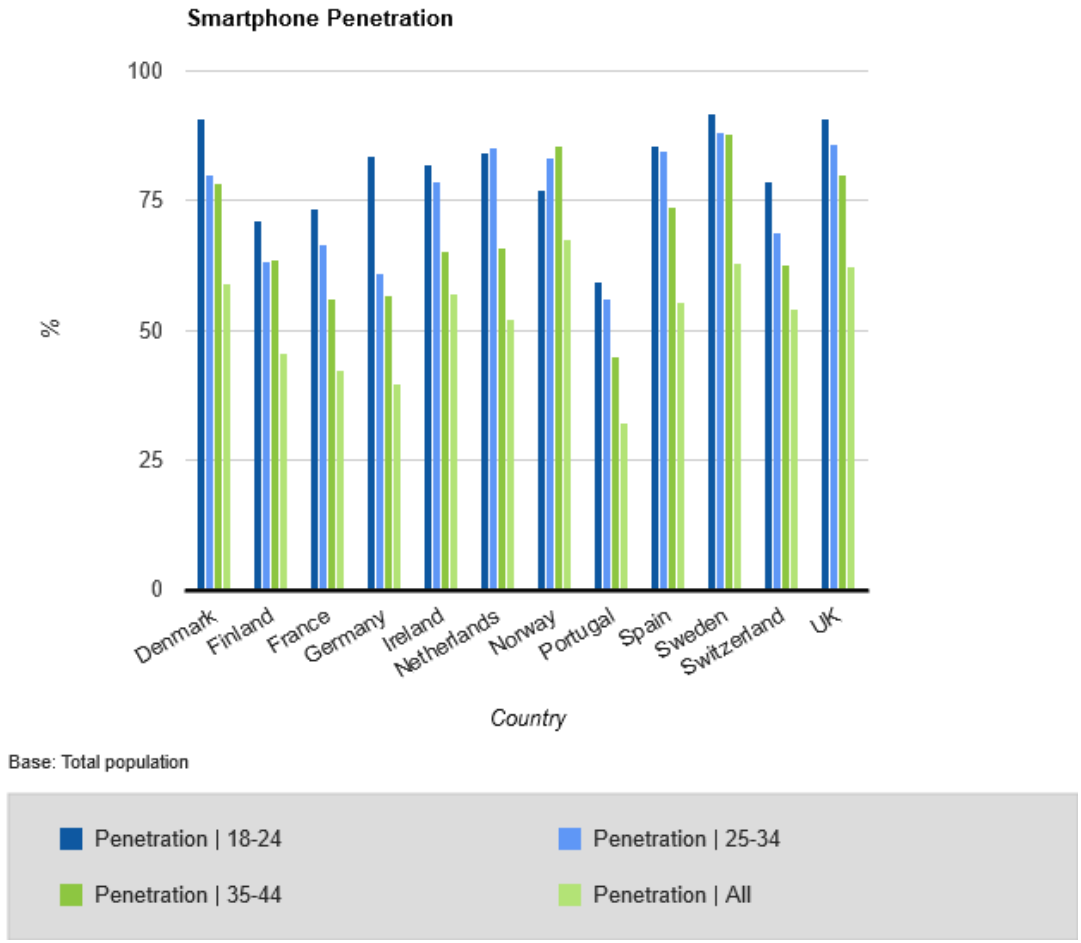

Fig. 1: Smartphone penetration in European countries depending on age range

[1] http://www.emarketer.com/Article/Worldwide-Smartphone-Usage-Grow-25-2014/1010920 [last accessed: 02/12/2014]

[2] http://think.withgoogle.com/mobileplanet/en/ [last accessed: 14/11/2014]

Smartphone penetration is generally bigger among young users between 18 – 24 years old, reaching in some countries almost 100%, but there is an exception in Norway, where penetration among users ageing between 35 – 44 is bigger than among users between 25 – 34 years old and users ageing from 18 to 24.

Note the case of Germany where 83 % of the population between 18 and 24 years old has a smartphone and despite of this big number the overall smartphone penetration is only 40 %.

In the case of Portugal is only 60 % and 32 % respectively.

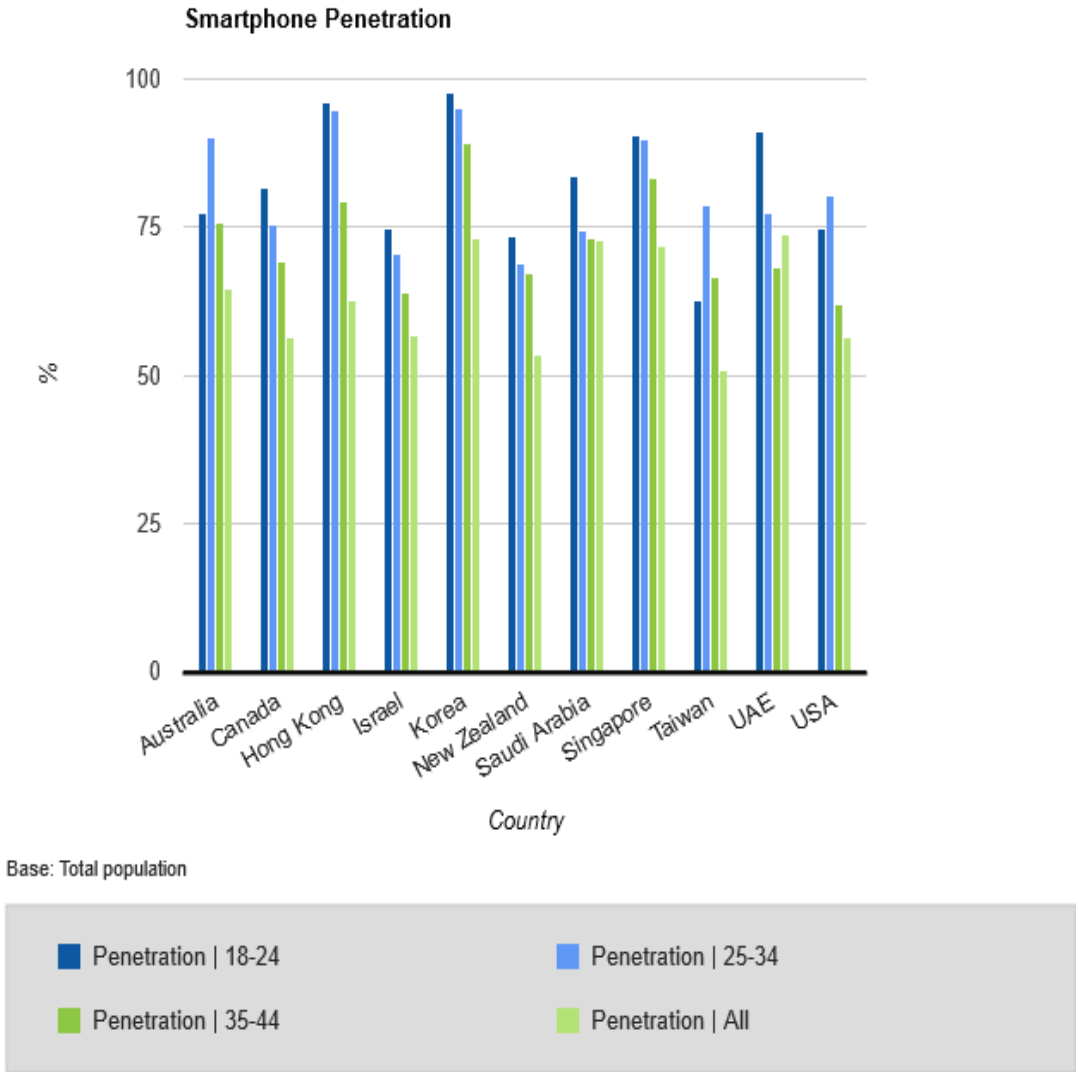

Fig. 2: Smartphone penetration in non European countries depending on age range

Outside Europe, is also a trend that smartphone penetration is bigger among users from 18 to 24 years old, but there are three countries where smartphone penetration is bigger among users from 25 – 34 years old than among 18 – 24 years old, and these countries are Australia, Taiwan and USA.

With this amount of users and the expected growth rate (25% in one year), more and more services and applications for smartphones are being developed every day. As the figures show the target population is basically young users, note that these figures do not include data for users under 18 years old, so a big percentage of younger users are not included.

Mobile applications have to be cautious regarding when and how much information is going to be provided to the user, if the information is not useful, or is provided too much frequently, the user will discard the application and uninstall it, around 25%[3] of the total apps fail in this and are uninstalled.
To avoid this overload of information, applications can check some facts and decide if it is the right moment to push some information or not, this facts could be for example:

- How long it has been since the user opened the application.
- How much time since the last notification was sent from the app to the user.
- If the user opens the notifications or ignores them.

Although these are good factors to decide if the user needs information or not, the most important factors are the current time, and the current geographical location of the user. This is the core concept of geofencing, thus, with these two variables applications can provide the information at the right place and at the right time.

To do so, applications have previously established a set of geofences, delimited areas, where the information to be delivered is important. The application has access to the GPS sensor or other source of information for the user location, and when the user enters one of these delimited areas, an event is triggered.

[3] http://mobiforge.com/research-analysis/global-mobile-statistics-2013-section-e-mobile-apps-app-stores-pricing-and-failure-rates#mobile-app-flops [last accessed: 02/12/2014]

In our context the event is to send a notification to the user mobile device, it is done by sending some data to a service, but this service could do any other thing with the data, for example just storing it in a database to track users location.

1.2.- Definition of terms

- Geofence[4]: It is a region delimited by boundaries. This boundaries can be circles or polygons and they correspond to a physical location.
- Geofencing: The technique using geofences.
- Geotrigger: Technique which trigger some information when a mobile device cross a boundary.
- Event / Action: In this context it is sending data to a server to perform some task as update database or send notification.
- Beacon: In this context it is a geographically localized sensor used to locate devices near to it.
- SmartCampus / SmartCity: It is a set of tools with geographically located data. These tools allow users to navigate, search offices, people or buildings and visualize information as power or water consumption in buildings.

1.3.- Motivation

Geofencing services allow applications to provide information in a more useful way, when the user is more likely to need it. Information is displayed at the right time and in the right place.

This a list of various use-cases for geofencing service:[5]

- Retail and Loyalty[6]: Stores may create loyalty cards and give prices to their customers, or send offers and personalized content when customers enter the store to engage them.
- Real Estate: Prospective home buyers may receive information when their search criteria match a home nearby.

[4] https://developers.arcgis.com/en/features/geotrigger-service/ [last accessed: 02/12/2014]

[5] https://developers.arcgis.com/geotrigger-service/guide/use-cases/#other-use-cases [last accessed: 02/12/2014]

[6] http://www.fastcolabs.com/3028819/the-big-question-in-retail-now-is-what-are-you-doing-with-geotriggers [last accessed: 02/12/2014]

- Energy Management: Users may use their location to automatically manage power consumption in their home or office.
- Tourism: Tourists could have information about nearby attractions, the interesting point is that this could be done offline, because most tourists do not have internet connection.
- Public Alerts: Citizens may receive alerts such as road closures or civic emergencies based on past locations.

There are some applications currently using geofencing techniques which many of them are really popular among smartphone users:

- FourSquare[7]: Allow users to do automatic check-in when they enter places and post comments and reviews about these places.
- Reminders and Notes: Is an app integrated in the core of iOS and since iOS 5[8], this app allows the user to set a reminder for a specific location. When the user enters that location the phone shows the reminder.
- Passbook and flight tickets[9]: Passbook is another iOS app integrated since iOS 6, it is location and time aware. It allows other apps to save tickets in it and they are shown on the lock screen when the device is in the right place and the right time.

Geofencing has many applications, some of them really surprising like for example in Kenia[10], where the elephants can get into the planted areas and wipe six months of income at a time, thus to protect both villagers and elephants, they delimited the crops with geofences, and installed SIM cards to the elephant collar which are able to send a text message to the guards whenever an elephant get close to these regions.

This geofencing technique is also very useful in the context of an SmartCampus or SmartCity tool where users navigate and search within the application to provide them with the right information in the right place. One SmartCampus tool, SmartUJI[11], is being developed in

7 https://es.foursquare.com/download#welcome [last accessed: 02/12/2014]

8 http://jeffreydonenfeld.com/blog/2011/10/appless-ios-5-finally-adds-geofencing/ [last accessed: 02/12/2014]

9 http://www.lufthansagroup.com/en/press/news-releases/singleview/archive/2012/september/21/article/2224.html [last accessed: 02/12/2014]

10 http://www.cbsnews.com/news/kenya-uses-text-messages-to-track-elephant/ [last accessed: 02/12/2014]

11 http://smart.uji.es/ [last accessed: 02/12/2014]

Universitat Jaume I. In this dissertation a possible integration with an environment of SmartCampus or SmartCity is provided, and the same technique could be applied and used in any of the virtual campus or cities retrieving and using geographical location data.

There are already many tools working with various data in SmartCampus, but none of them are using geofencing. Also information about air quality is not used in the system and we have the possibility to take samples of CO^2 data, therefore, to try this tool we will use real CO^2 concentration in air, but the same process could be applied to other values like electricity consumption, weather conditions or public alerts in buildings.

1.4.- Objectives

This dissertation aims to show the importance and possibilities of the geofencing technique to provide more accurate data to the smartphone users in an SmartCity context.
This tool will be used to provide notifications to the user's device when they are crossing previously established boundaries.
This tool will focus on providing information about CO^2 concentration in the place where the device is located.

1.5.- Dissertation outline

This dissertation is organized in seven sections. Section one contains the introduction, motivation and the objectives of this dissertation. In section two a state of the art is provided, also a table with comparison of some of the current solutions for geofencing is shown. In section three the methodology applied is explained. Section four shows the analysis and results. Section five contain the conclusions, and section six ideas for further work. Section seven contains all the bibliographic references used in the text.

2.- Literature review

Many applications are using services that require the user location to work, and the results or the information displayed is based on the geographical location of the user. These services are called Location Based Services (LBS).
Schiller and Voisard (2004) designated LBS as “Location services can be defined as services that integrate a mobile device's location or position with other information so as to provide added value to a user”.
Weather, navigational, shipping or tourist apps are some examples using LBS, but LBS is not used only in mobile applications.

LBS was a concept used before the smartphones were invented and Global Positioning System (GPS) was the first infrastructure serving the positioning of objects and people, it started working in the 1970's by the U.S. Department of Defence for military purposes, but in the 1980's the U.S. Government made the data freely to access by other industries.
Nowadays GPS is only one technique to locate users or objects, there are many other techniques as Bluetooth, infra-red sensors or other beacons, wi-fi, Assisted GPS (AGPS), Radio-frequency Identification (RFID), Zeimpekis et al. (2002) proposed a taxonomy for location techniques and divided them into two big groups:

- Self positioning: as GPS and AGPS, only valid for outdoor positioning, generally with high accuracy.
- Remote positioning: as Cell ID, using the cell the phone is connected to locate the user, works for indoor positioning, but has a very low accuracy (+- 200m).

Harter et al. (1994) proposed a system called the Active Office where the equipment and workers were located by using infra-red transmitters and badges, this system is one of the pioneers using LBS to construct location based heuristics for control and interaction among workers and equipment. The benefits they remarked as attractive are the physically non-intrusiveness of the system and the non-reliability on explicit user actions.
In 1996 the Federal Communications Commission (FCC) in the U.S. created the rules for wireless Enhanced 911 (E-911), the emergency number in the U.S., in the first phase of these rules, carriers must be capable of transmitting the originating number of the call as well as the location of the cell site receiving the call. In the final phase of E-911, completed in 2001, the carrier must provide latitude and longitude of the caller's position within an accuracy of 125m.

Similar rules were applied in Europe with 112 number and 999 number in UK. Reed et al. (1998) explained the challenges to meet these rules and describe various positioning methods with their advantages and disadvantages considering scenarios such as urban areas or rural areas.

Retrieving the geographical location from the user of the mobile device is a key feature that all these applications use, and localize the user not only in a geographical location, but also in a context, the final purpose of these applications is very different, but all of them make use of user geographical location:

- Google Latitude[12]: Used to share your location with your friends, see where are your friends and who is closer to you. This service was integrated into the Google Maps application, but was cancelled in August 2013.[13]
- Find My Friends[14]: By Apple, same service as Google Latitude but also allows to set notifications of events, such as one of your friends left or entered one place.
- Nearby Live[15]: Works as a social network, users create a profile and they can chat with nearby users to meet new people.
- Ingess[16]: By Google, uses the real world as the landscape for a global game of mystery and intrigue where each player must choose a side.
- Mosey[17]: Users save their favourite places in a city, and other users can search by location and visit these places.
- Groupon[18]: Users can search and buy discounts for products or services located near them.
- Assistive Health Care: Pourhomayoun et al. (2012) Describe a system to monitor and track patients with wireless sensors in indoor locations such as their houses.
- Location Labs[19]: Users can install this application to track the position of other devices and get notifications when they reach the places previously set with

[12] http://googleblog.blogspot.pt/2009/02/see-where-your-friends-are-with-google.html [last accessed: 02/12/2014]
[13] https://support.google.com/gmm/answer/3001634?p=maps_lat_faq [last accessed: 02/12/2014]
[14] https://itunes.apple.com/us/app/find-my-friends/id466122094?mt=8 [last accessed: 02/12/2014]
[15] http://www.wnmlive.com/apps [last accessed: 02/12/2014]
[16] https://www.ingress.com/ [last accessed: 02/12/2014]
[17] https://www.mosey.com/ [last accessed: 02/12/2014]
[18] http://www.groupon.com/mobile [last accessed: 02/12/2014]
[19] http://www.location-labs.com/ [last accessed: 02/12/2014]

geofences. Also they provide tools as to lock the screen of the phone when the user is driving and motion is detected to avoid driving hazards.

- Placecast[20]: Enterprises can place their advertisements in specific places with location algorithms, these ads are delivered to nearby users, which meet the requirements according to their audience profiling.

There are plenty of examples of other applications using user location, nowadays every social network can retrieve the user location to check-in, or simply to post user location.

The main issues with using user location in mobile devices are privacy issues, and the battery drain caused by the GPS sensor.
All LBS running on mobile devices using GPS prevent Android and iOS devices entering into the sleep state. When devices are in that state battery consumption is minimum, but with GPS sensor communicating to retrieve user's location a lot of battery power is spent. This is the reason why so many applications have been redesigned to decrease the number of times that accessing the GPS is needed. Nowadays there are many examples of applications that have been designed in order to take into account the battery power consumption and in this way increase battery life.

There are various techniques to avoid unnecessary usage of GPS, for example Zhuang et al. (2010) implemented and tested a framework on an Android prototype that follows four principles to make GPS access more efficient when there are some location based applications that use asynchronously the GPS, thus making the system more energy-efficient. In mobile platforms, many applications require the use of the GPS, but by default, there is no cooperation among them, also most of these applications are not aware of the battery level, and they used these limitations to improve the GPS access.
Their four principles are, substitution to make use of alternative positioning methods, suppression to use less power-consuming sensors such an accelerometer to determine when the user moves, and do not update the location if the user did not move, instead of updating GPS in a fixed time interval, piggybacking to synchronize the positioning requests from multiple running applications and adaptation to adjust sensing parameters such as time and distance when battery level is low. Their prototype reduced number of GPS invocations by up to 98% and improved battery life up to 75%.

[20] http://www.placecast.net/ [last accessed: 02/12/2014]

Paek et al. (2010) describe another positioning system, which uses the location history stored in the device by default to estimate the user velocity and turn GPS only when the estimated uncertainty in user location exceeds an established threshold, by prototype testing in a modern smartphone their results show that this system can increase phone life by more than a factor of 3.8 over a system where GPS is always on.

Bulut et al. (2013) describes the design, implementation and deployment of a crowdsourced line wait-time monitoring service. It consists of a mobile client and a service stored in the cloud, the authors describe two methods they are using to provide updates about user localization, avoiding as much as possible accessing the GPS sensor since this action is the one that consumes more battery power. In order to save as much battery as possible, their application tries to access first the last known location, stored in the device each time any other application needs an update of localization, and calculates time to access again the mobile stored location based on the distance to the place to monitor. The second method is using the place's wi-fi, they detect when the device first reach wi-fi and when it stops reaching the signal, the difference of that two timestamps is the waiting time.
In the second method the wi-fi is acting like a geofence which physical boundaries are the actual wi-fi signal coverage.

Geofencing techniques are also highly related to security and privacy issues in location based applications, since these techniques can be applied to restrict information or wi-fi coverage to a physical boundary.
There are some cases where wi-fi should be confined to specific places, as conference rooms, libraries or coffee shops, but wi-fi signal can cross walls and usually does not have the place shape, thus people who are not in the place can receive the signal. Sheth et al. (2009) explain how to increase security and privacy of wi-fi using geofencing.
They implemented a system that combining the geofence technique with wi-fi directional antennas can constraint wi-fi coverage to physical boundaries or regions from a single desk or a large room. With this systems clients can be authorized to use the wi-fi simply by being inside the building with no need for passwords.
But there are more technologies that can act as a geofence apart from wi-fi, for example Ijeh et al. (2010) implemented a system using geofencing techniques with Radio Frequency Identification (RFID) chips and demonstrate how to turn their organization wireless connection from insecure to highly secure.

There are many examples of location based applications that use some sort of geofencing techniques to provide information in the right place and even at the right time.

Ludford et al. (2006) describes the implementation and testing of a location based reminder application, this application allows users to place reminders in specific places to help them with their everyday life, these reminders are targeted to themselves. When saving the reminder in the system, users fill the description, the place they want it to be delivered, and optionally a time and date. Each reminder is sent to their user via their mobile phone application when users are within a predefined radius from the place, or alternatively if they are moving, the application calculates distance to the point and sends them a reminder when the user is within a predefined time to arrive.

Espinoza et al. (2001) studied the social aspects of location-based information systems and describe a system to provide information like post-its, graffiti and posters does, this information is attached to one physical location and can be addressed to all users, to specific users or to groups of users such as user's friends. The system is called GeoNotes and it allows users to post notes in one localization. In a system like this, information needs to be limited and structured to avoid overcrowded locations or not useful information improving usability of the system. This is the reason why they organize locations into tree structures that describe the real world as locations within locations, such a campus has buildings, a building has departments and a department has rooms. Also, they suggest user clustering could be applied based on some machine learning techniques about the actions performed in the system. As a sorting mechanism to give priority to the relevant notes, they offer sorting by recent notes, the ones addressed to the user, or based on the popularity of the note. This popularity is increased when a user saves the note, and is decreased when a user ignores it.

Küpper et al. (2011) describe the two generations of Location Based Applications (LBA) that have been created in the past decade. The first generation of LBA was basically providing information through SMS and the GPS was still not available on mobile devices. From 2007 it starts the second generation of LBS, GPS and other positioning systems allow users to share their location, share it, check in at certain places, and a broader range of functions for LBA are created. As future trend they forecast geofencing and background tracking are going to be used more in LBA due to the new multitasking capabilities on mobile devices, these allow mobile devices to send current location to the servers when the application is running in the

background or even when the application is terminated. As the current limitations of these new techniques they mention the battery consumption problem and privacy issues related to location that could be used by attackers to track user's favourite places or costumes.

3.- Methodology

3.1.- Overview

This system is constituted by a CO^2 sensor, two services and the mobile devices as clients. We had the opportunity to access one CO^2 sensor to gather real data. Then the geoprocessing service, using interpolation to cover all the area, defines which areas have more CO^2 concentration and the geofencing service creates geofences from these areas. Clients will receive a notification on their mobile phones when they are crossing one of these areas.

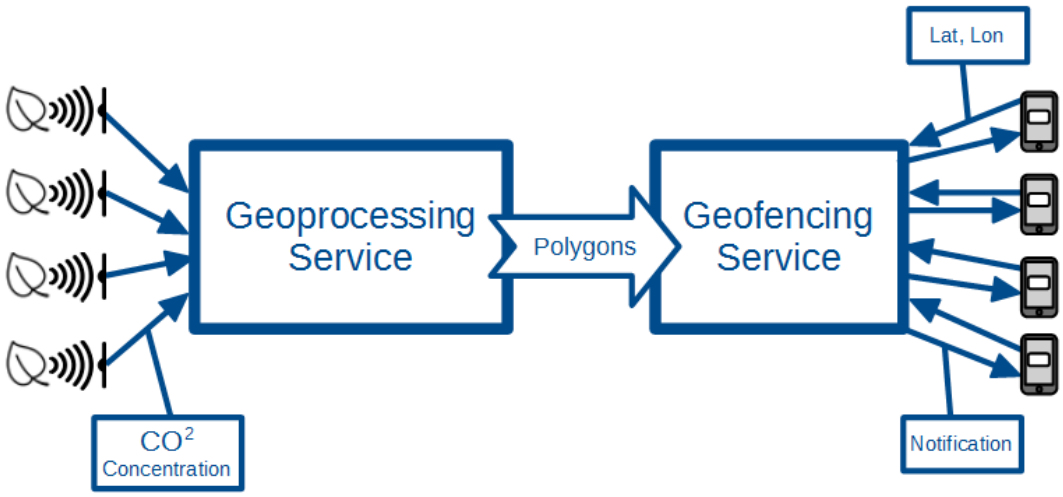

Fig. 3: Overview of the system

3.1.- Sensor Data Acquisition

We had access to only one CO^2 sensor, so at this moment the data is not in real time.
The CO^2 sensor was placed in some locations in the city of Castelló de la Plana in Spain.
Data samples were collected during the weekdays, due to the fact that only one sensor was available to take samples, we assumed that weekdays at the same hour will have almost the same CO^2 concentration.
In our readings CO^2 concentration ranges from 260 to 410.
The sensor writes the values in the following format:

year#month#day#hour#minute#second#CO^2#temperature

These are the readings for 17th January 2015:

2015#1#17#7#30#41#397#5

2015#1#17#7#35#34#408#4

2015#1#17#7#40#28#404#4

2015#1#17#7#45#22#388#4

The sensor is programmed to read data every 5 minutes, so if more sensors were available the service could update the sensor data and the regions could be updated in real time.

3.2.- Geoprocessing Service

To make use of the geofencing technique we will need a service that creates and updates the regions where we want to notify to the users the required information, in this case the CO^2 concentration.

For this purpose ArcGIS software was used, because we can publish the sequence of steps performed on the ArcGIS server as a geoprocessing service, and then access it when we need to retrieve updated data. Moreover, all the tools related to SmartCampus in UJI are using ArcGIS technology and are stored in ArcGIS servers, in this way the cooperation between the tools will be easier and faster.
The first basic step in this service is to take the point data and perform an interpolation to cover all the areas that do not have CO^2 data. The second step is to convert the resultant raster image to a set of polygons.

We stored in a geodatabase one layer containing the locations where the sample data was taken and each point contains the CO^2 concentration read in that location. The geodatabase is published in the server, thus we can update the values when we get new CO^2 readings.

Junninen et al. (2004) tested some methods to fill the missing values in air quality data, the method used gave very different results, but the result also depends on how many samples exists and the distance between them. Of course, their study area is US, and is not comparable in size with our study area, which is a city with less than 200.000 inhabitants. But the method we used is also interpolation.

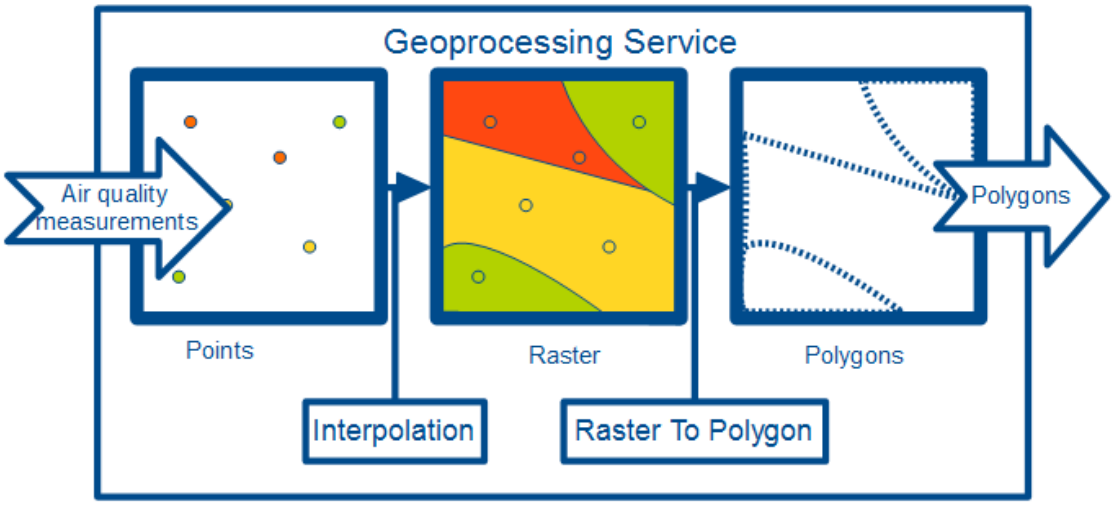

Fig. 4: Geoprocessing Service diagram and the steps involved in it.

There are some methods to do the interpolation[21], generally they can be classified into two big groups[22]:

- Deterministic: IDW (Inverse Distance Weighted), Natural Neighbour, Trend, and Spline. These methods assign values to locations based on the surrounding values and on mathematical formulas that determine the smoothness of the resulting surface.
- Geostatistical: Kriging. These methods are based on statistical models that include autocorrelation. Because of this, geostatistical techniques also provide some measure of the certainty to predict errors.

Sluiter (2008) classified various methods in the following categories:

- Local or Global: Global interpolators use a single function to map the entire dataset, and local may use various functions on subsets within a predetermined window.
- Exact and Approximate: In exact interpolators the location of the sample points have the same value in the result than in the attribute interpolated, approximate interpolators assume uncertainty in all points and values may not be the same.

[21] http://webapps.fundp.ac.be/geotp/SIG/interpolating.pdf [last accessed: 25/01/2015]

[22] http://help.arcgis.com/en/arcgisdesktop/10.0/help/index.html#/An_overview_of_the_Interpolation_tools/009z00000069000000/ [last accessed: 25/01/2015]

In our context, we do not need to do accuracy assessment, because the polygons will be reduced, so anyway, some data will be lost, thus we will use deterministic method. Also, is better to use an exact interpolator, as IDW or Spline, to maintain the same values where the samples were taken, regarding the local or the global method some tests were performed to choose the most convenient method for our purpose.

To perform a realistic interpolation about air quality data as Junninen et al. (2004), we are missing some data that interferes with our samples, as wind speed, wind direction, temperature and humidity, but our goal is not to create a model for CO^2, so all these variables are not taken into account.

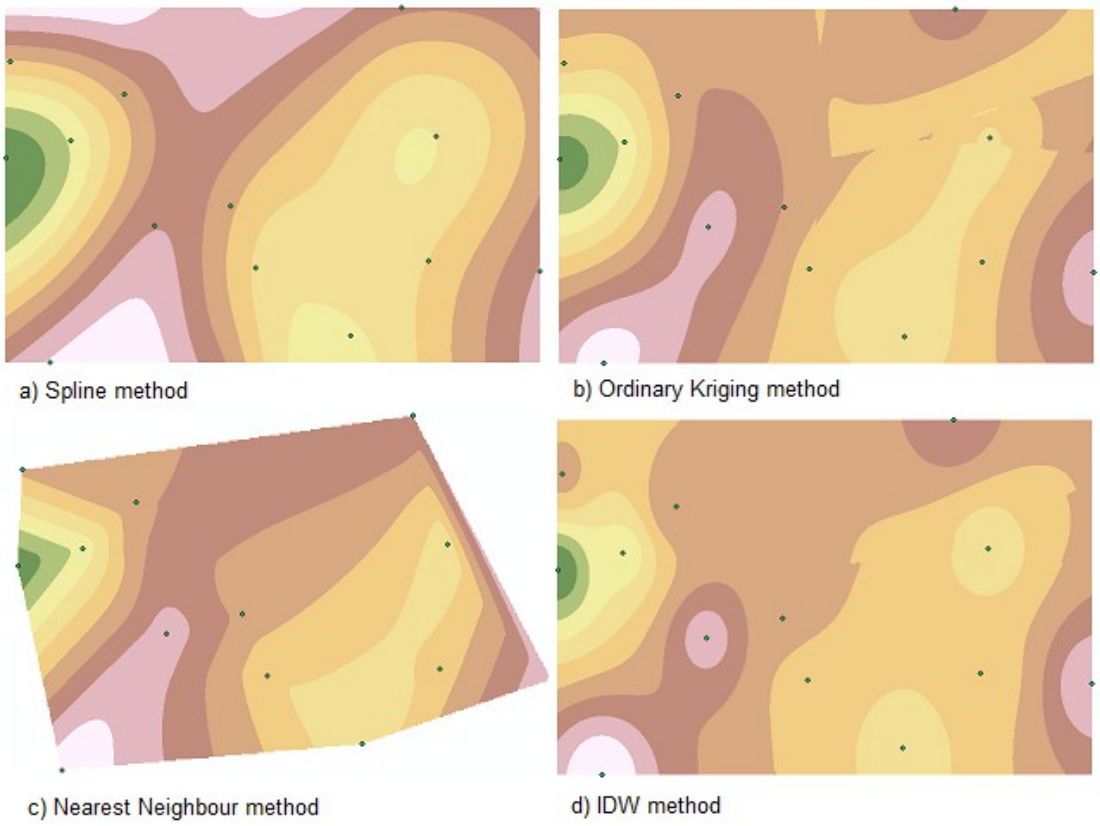

Fig. 5: Comparison of various interpolation methods

Wong et al. (2004) compared some interpolation methods for air quality data, we will test the same methods: IDW, Kriging and Nearest neighbour plus Spline.

The comparison suggest that IDW is the one that shows better results for our goal, to create polygons from it, plus it is a deterministic and exact interpolator, so we will choose IDW for our model.

After doing some tests with the available tools in ArcMap we built a model, which takes inputs, in our case the sensor points, and generates an output.

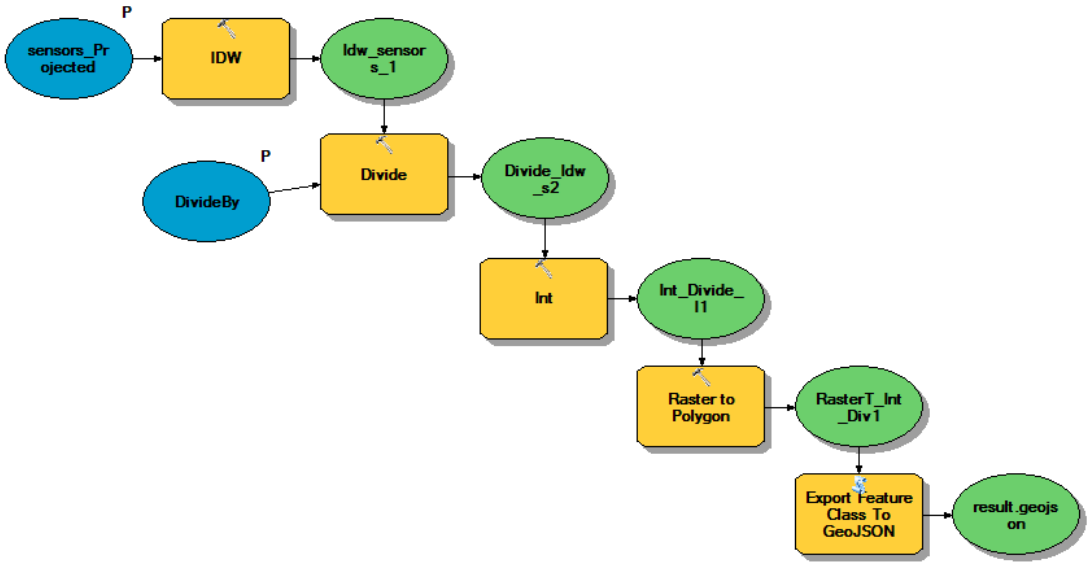

Fig. 6: Model in model builder. Inputs are represented in blue colour, tools in yellow and outputs in green colour.

We will run this sequence of tools to reach the final result:

1.- **IDW**: Interpolates a raster surface from sample points.

2.- **Divide**: We need this tool in order to reduce the number of polygons created. In our context, we do not need high accuracy, but rather well defined polygons. The model takes as parameter the number to divide with, so the final result will have more or less polygons depending on which number is passed as parameter.

3.- **Int**: This tool is used to remove the decimal part of the data sample, in order to be able to run the next tool.

4.- **Raster To Polygon**: This tool takes as input the raster and converts the areas with the same values to polygons.

6.- **Feature Set To GeoJSON**[23]: This python script outputs to a file the resulting polygons in a GeoJSON[24] format. GeoJSON is a format to encode geographic data structures, including polygons in a text file, is easy to read and is widely used, it follows the same syntax as the JSON[25] format. The resultant polygons in the GeoJSON format look like this:

[{"geometry":{"type":"Polygon","coordinates":[[[-0.040868,40.000633],[-0.023961,40.000633],
[-0.024200,40.000375],[-0.025634,39.999276],[-0.027278,39.998235],[-0.029350,39.997248],
[-0.030775,39.996704],[-0.032146,39.996303],[-0.033871,39.996002],[-0.035902,39.996119],
[-0.037206,39.996531],[-0.038325,39.997162],[-0.039288,39.997918],[-0.040103,39.998981],
[-0.040497,39.999727],[-0.040868,40.000633]]]}, ….

As we can see the last coordinate is the same as the first one, which means the polygons will be always closed, and we can use them to build geofences.

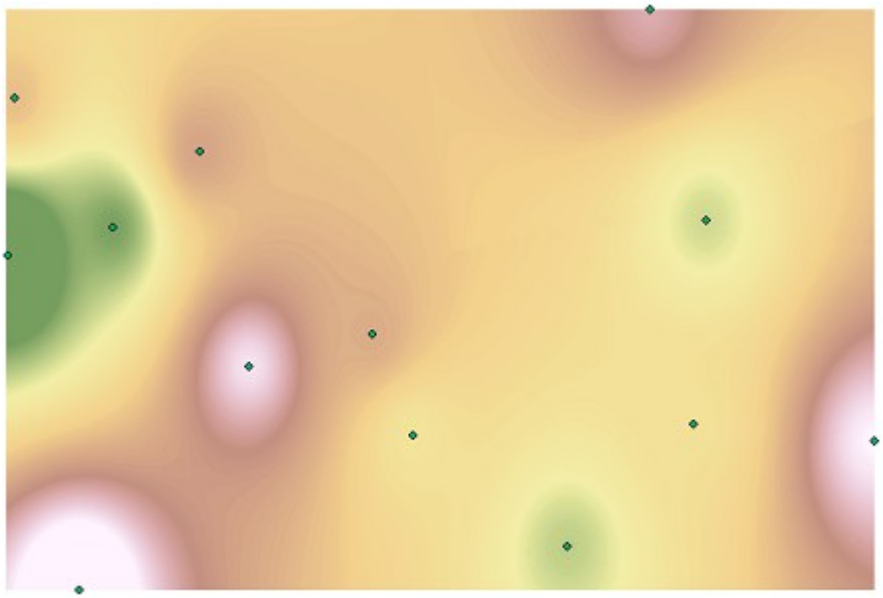

Fig. 7: Raster resulting from IDW interpolation (Step 1)

[23] http://arcscripts.esri.com/details.asp?dbid=15545 [last accessed: 22/01/2015]
[24] http://geojson.org/ [last accessed: 22/01/2015]
[25] http://json.org/ [last accessed: 22/01/2015]

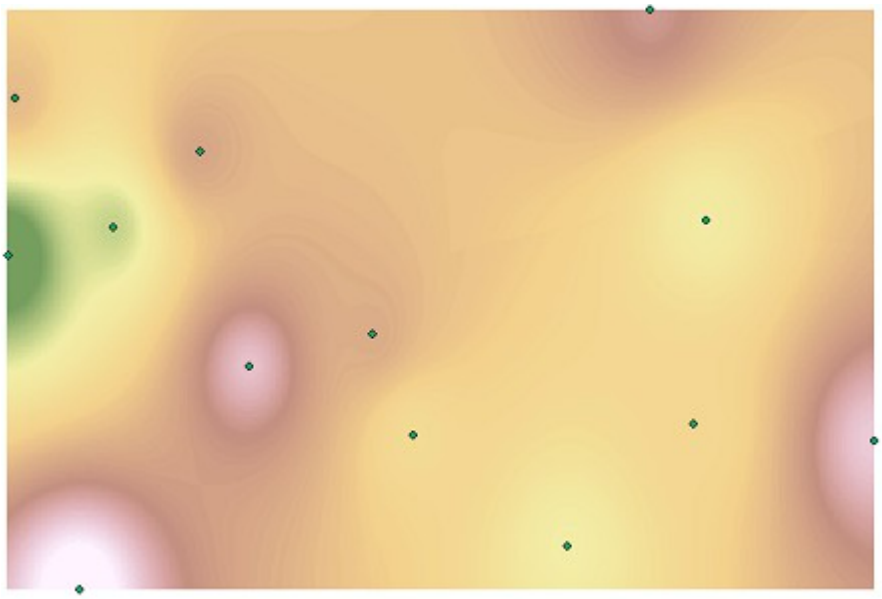

Fig. 8: Raster resulting from dividing the previous raster (Step 2)

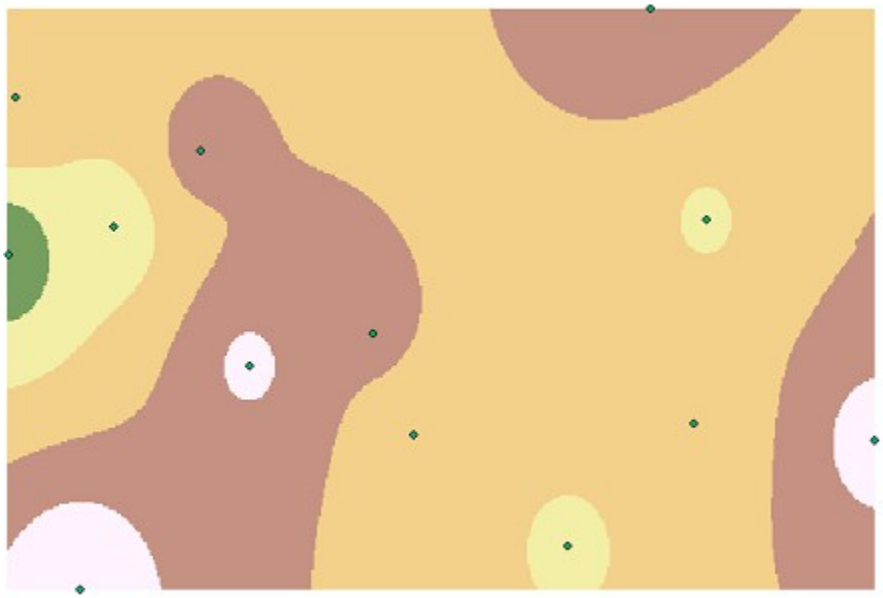

Fig. 9: Raster resulting from converting to integer the previous raster (Step 3)

We can use an online tool as geojson.io[26] to visualize the GeoJSON file created by the model.

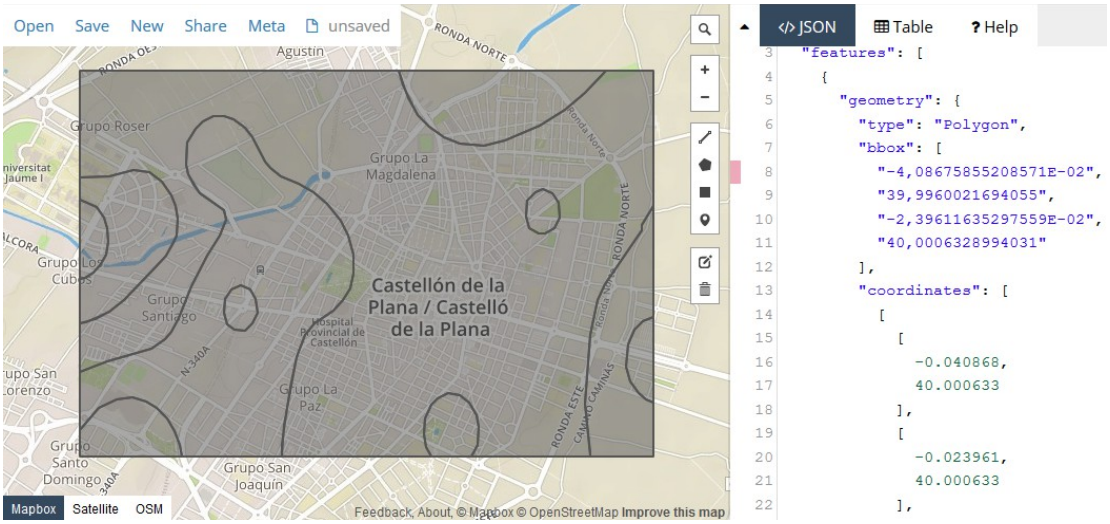

Fig. 10: Visualization of GeoJSON file created by the model in geojson.io online tool

After the GeoJSON file is created, our second service will read the polygons and create geofences, but create the geofences will depend on which geofencing solution are we using for the client application. There are some possible methods and each of them is using their own framework and syntax. That is why the next point is the client application and the one after is the second service, the geofencing service.

3.3.- Geofencing Client

Although geofencing technology is a pretty new concept first used only in the native Android or the native iOS operative systems, nowadays there are some companies that offer geofencing management at a higher level than adding programmatically the geofences in the code.

To give a broad vision on the current solutions, and what advantages do they offer compared to the native region monitoring a comparison of various methods is given.
The proposed methods are:

1. Native region monitoring

[26] http://geojson.io [last accessed: 22/01/2015]

2. ESRI geotrigger service[27]
3. Proximity Kit geofencing framework[28]
4. Plot Projects geofencing framework[29]

All of them could be tested in both Android and iOS operative systems, but due to the limitation in time and resources, iOS operative system was chosen to make the tests.

In the next section more complete information about how to use each method integrated in iOS application is given.

A research on the state of the art of this technology was done, including the native and the ESRI approaches plus other two solutions found on the Internet were considered to be analysed and tested in the mobile application. The main goal in this testing is to detect which approach has less impact on the battery consumption, which is directly connected to the number of times that the application request data from the GPS sensor.

[27] https://developers.arcgis.com/en/features/geotrigger-service/ [last accessed: 02/12/2014]
[28] http://proximitykit.radiusnetworks.com/ [last accessed: 02/12/2014]
[29] http://www.plotprojects.com/ [last accessed: 02/12/2014]


	1. iOS Native	2. ESRI geotrigger	3. Proximity Kit	4. Plot Projects
Provides interface to manage geofences?	No	Yes + Usage analytics	Yes	Yes + Analytics
Free account for developers	Not required	600 geotrigger events	5 geofences 100 Active Installs	10 geofences Unlimited notifications
Documentation	Very good	Very good	Short	Good
Platform support		  + Web	 	  + App generators

Table 1: General characteristics of various geofencing methods

To test these applications in iOS device is mandatory to use Xcode[30], the Integrated Development Environment (IDE) to build and test iOS applications. With Xcode a set of iPhone and iPad simulators are installed to test our application in various devices, but the final test was done in the real device to track battery consumption.

Xcode contains also a set of instruments that record the performance of the application, to detect memory usage, CPU usage, zombie objects and many other tools.

[30] https://developer.apple.com/xcode/ [last accessed: 23/11/2014]

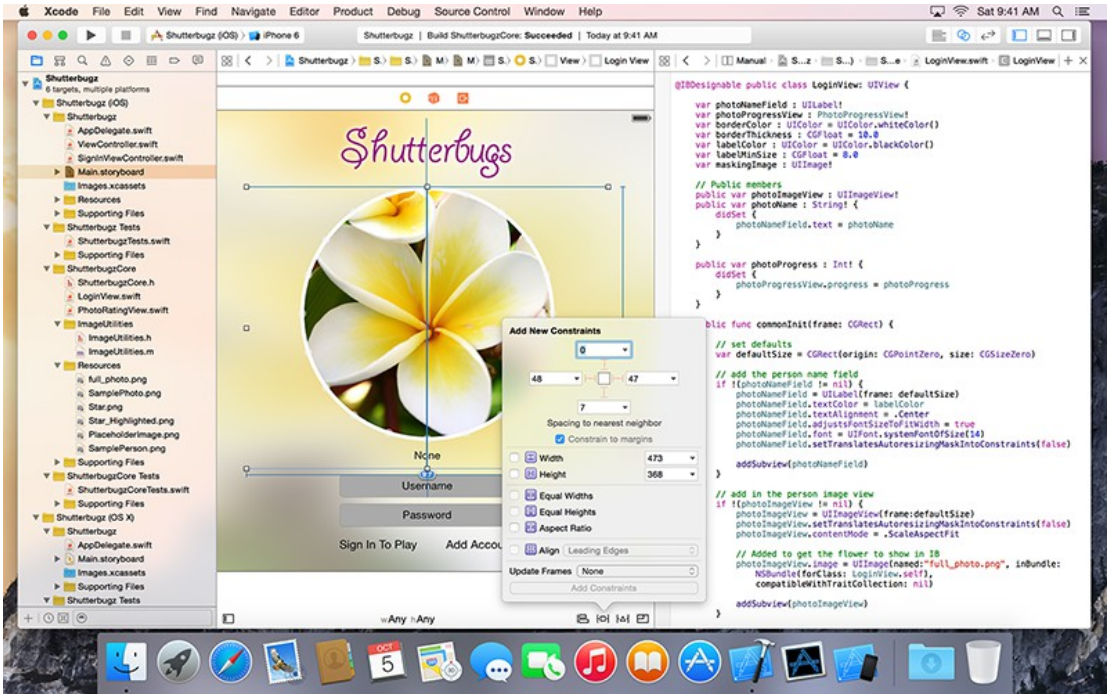

Fig. 11: Main screen of Xcode 6, IDE to build iOS 8 apps

The programming language for iOS development is either Objective-C[31] or the recently introduced Swift[32]. Although Swift claims to be easy and quicker to write, is still in an early phase and has errors to be solved hence the programming language used in the mobile applications is Objective-C.

A simple application was implemented for each of the proposed methods, in the following sections the development of each approach is explained in more detail.

3.3.1.- iOS native region monitoring

To use the native region monitoring is necessary to add the CoreLocation framework, this framework is added directly from the Xcode and allows the application to access the device location information, including GPS readings and location history.

[31] https://developer.apple.com/library/ios/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/Introduction/Introduction.html#//apple_ref/doc/uid/TP40011210 [last accessed: 23/11/2014]

[32] https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/index.html#//apple_ref/doc/uid/TP40014097 [last accessed: 23/11/2014]

3.3.1.1.- Creating geofences

In the native approach there is not a graphical tool to create geofences and it is done programmatically.
First is necessary to declare and initialize an instance of CLLocationManager, which is the object that contains all the device location and the functions to work with the region monitoring.

```
self.lm = [[CLLocationManager alloc] init];
```

Since iOS 8 applications have to request permission to access the device location data, in our case we need the location data even if the application is in the background, therefore we request authorization for always.

```
[self.lm requestAlwaysAuthorization];
```

After having the permission from the user the application starts creating the regions, these regions are created from a 2D coordinate, latitude and longitude, a radius, which will determine the extent of the region and an identifier.

```
CLLocationCoordinate2D isegiCoor = CLLocationCoordinate2DMake(38.731896,-9.160103);
CLCircularRegion *geoRegion = [[CLCircularRegion alloc]
                               initWithCenter: isegiCoor
                               radius: 0.01
                               identifier:@"ISEGI"];
```

The CLLocationManager instance is the object which will start monitoring the region.

```
[self.lm startMonitoringForRegion:geoRegion];
```

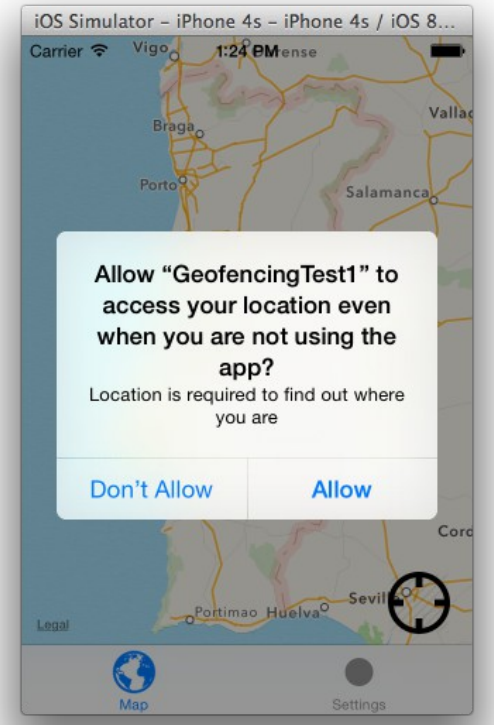

Fig. 12: Application asking for user permission to access location in background state

3.3.1.2.- Monitoring geofences

Two functions from the CLLocationManager allow the app to monitor when the device enters or exits one of the regions previously specified.

```
-(void)locationManager:(CLLocationManager *)manager didEnterRegion:(CLRegion *)region{
    NSLog (@"Device entered region %@", region.identifier);
    [self sendAlert:@"Enter" andMessage:[NSString stringWithFormat:@"Device entered
region %@", region.identifier]];
}
```

```
-(void)locationManager:(CLLocationManager *)manager didExitRegion:(CLRegion *)region{
   NSLog (@"Device exit region %@", region.identifier);
    [self sendAlert:@"Exit" andMessage:[NSString stringWithFormat:@"Device exit region
%@", region.identifier]];
}
```

This two functions print in the log of the system which region was entered or exit and send an alert to the device with the region identifier.

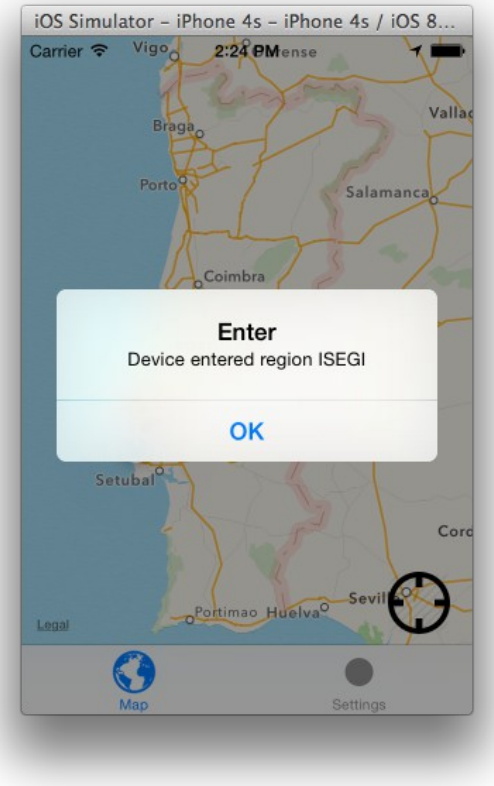

Fig. 13: Warning when device enters region

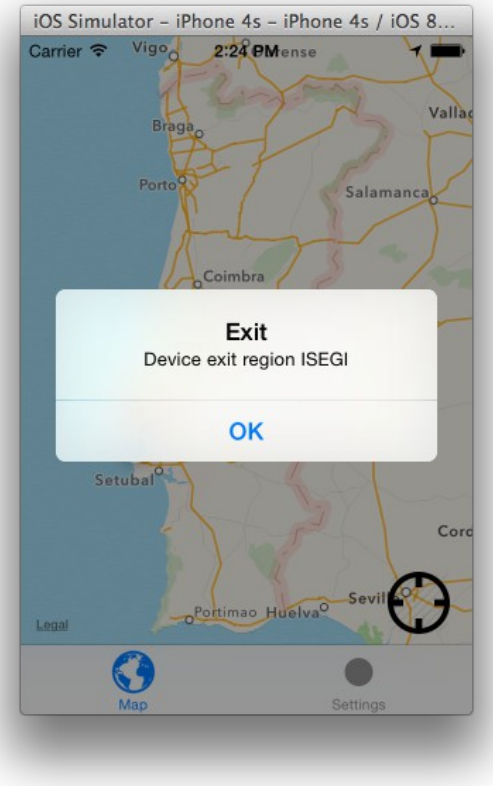

Fig. 14: Warning when device exits region

In the iOS simulator and devices is possible to simulate locations, adding files into the project with the desired latitude and longitude, and selecting it from a list.
These files have to be in GPX[33] (GPS eXchange) format, which is XML with GPS information:

[33] http://www.topografix.com/gpx.asp [last accessed: 25/01/2015]

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<gpx version="1.1"  creator="gpx-poi.com">
<wpt lat="38.731896" lon="-9.160103">
<time>2014-10-27T20:31:33Z</time>
<name>isegi</name>
</wpt> </gpx>
```

With these files is easier to test applications, the device physical location may be the same, but it updates the GPS data to the location in the selected file.

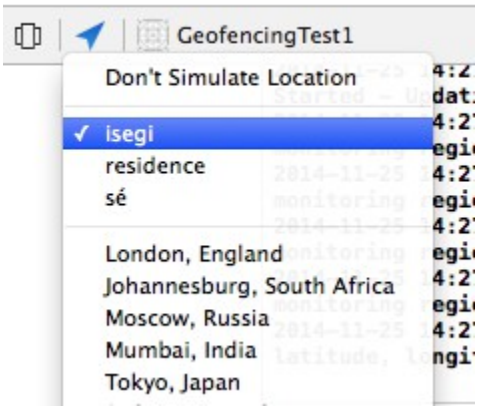

Fig. 15: Menu to choose what location to simulate

3.3.1.3.- Monitoring battery life

Devices have a variable where you can access information related to the current device, screen size, software version and battery level among others.

```
float batteryLevel = [UIDevice currentDevice].batteryLevel;
```

The battery level is printed to the console every time the level changes, and the output to the console is saved in a file inside the device. With this output we will be able to determine which method consumes more battery.

3.3.2.- Region monitoring with external SDK

To use an external SDK, like the ESRI geotrigger service it is necessary to include it in the project, sometimes there is some additional configuration, like including a plist file with loggin data or requesting an application ID to authorize it.

3.3.2.1.- Creating geofences

External software provide a graphical tool to draw geofences, all of them allow to draw circles, one centre coordinate and a number for the radious, ESRI allow also to draw polygons.

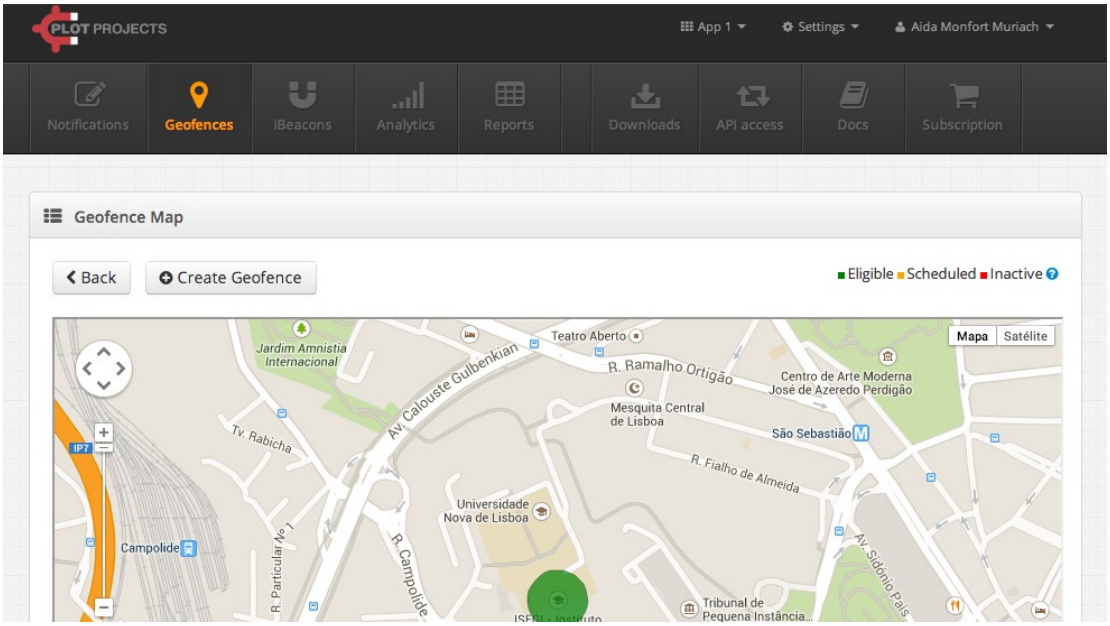

Fig. 16: PlotProjects interface to draw geofences

3.3.2.2.- Geofencing analytics and monitoring

The SDK monitors automatically the geofences crossed by the device and manage the warnings. ESRI and PlotProjects also provide information about the usage. There is also an ESRI tool called geotrigger faker, where you can simulate the behaviour of the SDK by giving it a simulated location and it returns in response which geotriggers are affected.

Monitoring the battery life is done in the same way as in the native approach.

3.3.3.- Geofencing client selected

The geofencing client we will use is the ESRI SDK, due to the fact that we need geofences to be polygons, also they manage automatically push notifications, which not only makes the user aware of the warnings, but also allows the application to run in the background and save battery life. Another advantage that the ESRI solution has is that we can use their API (Application Programming Interface) to manage the geofences by a service, there is no need to update the geofences in the graphical tool if we can use the API.

3.4.- Geofencing Service

Our second service will take the GeoJSON file created containing the polygons and build geofences.

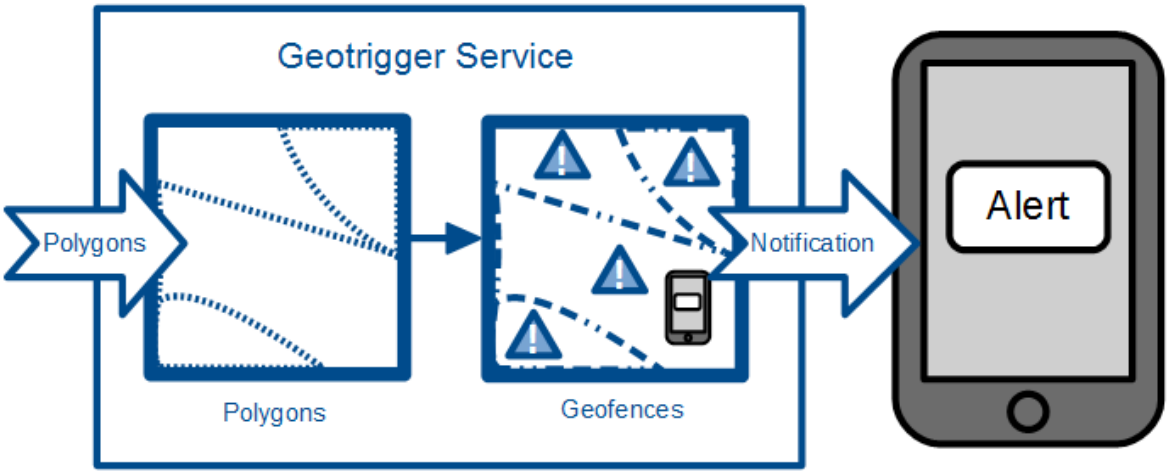

Fig. 17: Geotrigger Service diagram

To reach this goal, we could use our own service, but as ESRI provides an API[34] to create geofences and we will use it.

We will need a service to make all the process automatic, this service is going to read from the GeoJSON file each polygon and is going to make a curl[35] request, sending the GeoJSON file as parameter, to the API to create the geofences.

[34] https://developers.arcgis.com/geotrigger-service/api-reference/trigger/ [last accessed: 26/01/2015]
[35] http://curl.haxx.se/docs/httpscripting.html [last accessed: 26/01/2015]

```
// request
curl -H "Authorization: Bearer <Application or Device Access Token>" \
   -H "Content-type: application/json" \
   -d @trigger_create_2.json \
   "http://geotrigger.arcgis.com/trigger/create"
```

Fig. 18: cURL request to create geofence from a json file

After running this service we can access the geotrigger web page and check the updated geofences. The geofences will have tags associated, these tags will allow users to subscribe to them to filter the notifications they want to receive.

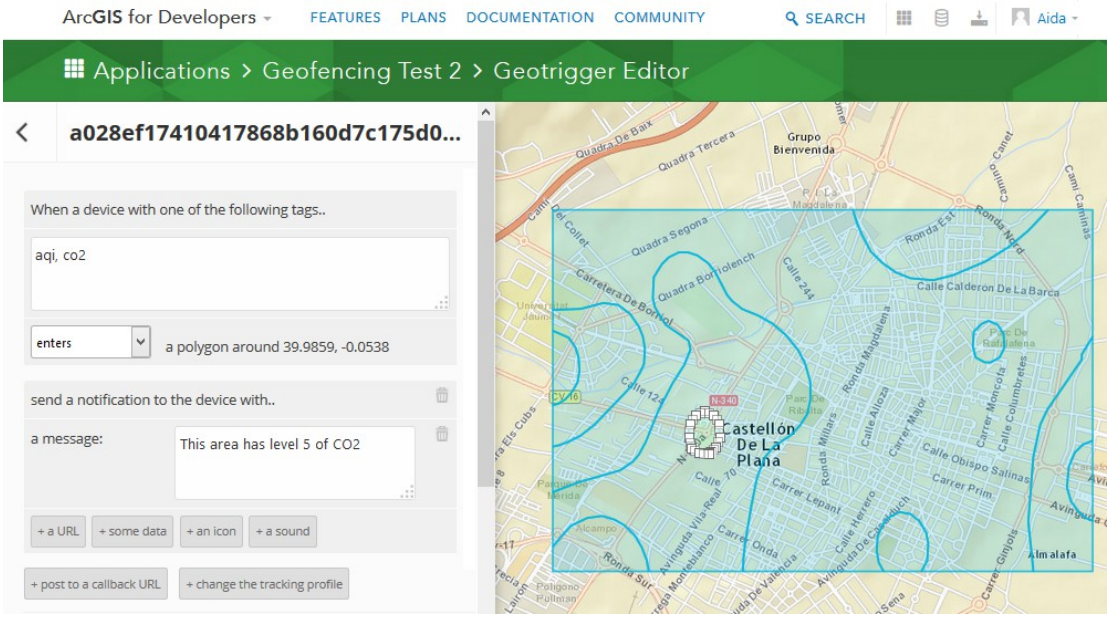

Fig. 19: Geotriggers created from GeoJSON file with geotrigger API

4.- Analysis and results

The main goal of the geoprocessing service is to provide some polygons with an attribute which is interesting for the users, and the geotrigger service should be able to send notifications when user devices enters the defined areas. In our case we used CO^2 concentrations in the air, as useful information for the users.

4.1.- Geoprocessing Results

The number of polygons created in the first test was too much and they were too close one from each other, testing the client application, bests results are when geofences are larger than 50 m radius, and in the first result polygons were 20 meters or less one from each other.

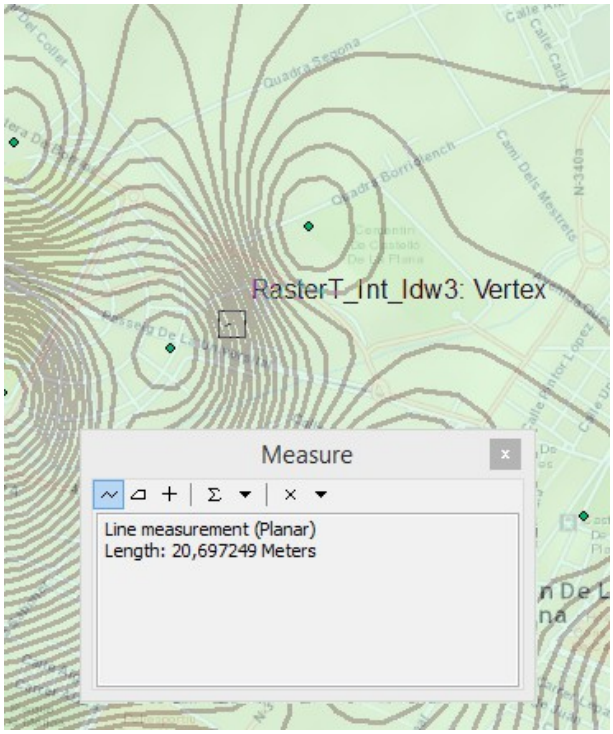

Fig. 20: Polygons created before dividing values

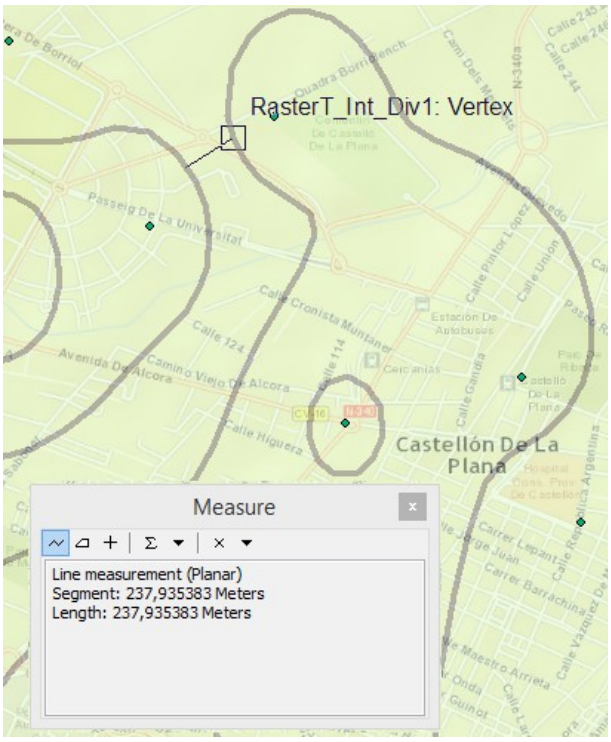

Fig. 21: Polygons created after dividing the values

After dividing the values by 10, the polygons were less and less close one from each other, the minimum distance is about 240 m, which make sense in a city with Castelló size. With this polygons geofences will work fine.

4.2.- Testing region monitoring results

When testing LBA, there are some issues to take into consideration, if the device is connected to Wi-fi the accuracy of the location is improved, however, in this test applications were tested with no Wi-fi, only GPS and 3G connexions which is the common situation when users are using LBA. Also GPS signal takes few seconds to retrieve the correct data, in the worst case it can take up to 12.5 minutes[36], but mobile devices use AGPS which works with estimated data, and updates when more accurate data is received. In the maps included in mobile applications the estimated data error is drawn with a semi-transparent blue circle.

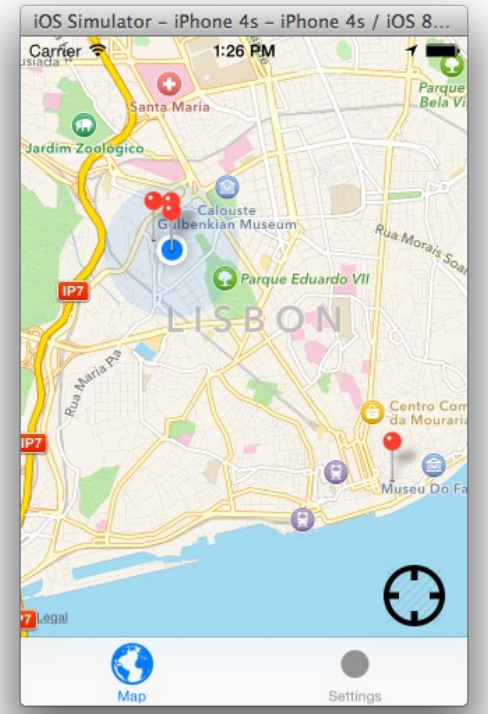

Fig. 22: Low location accuracy immediately after launching the application

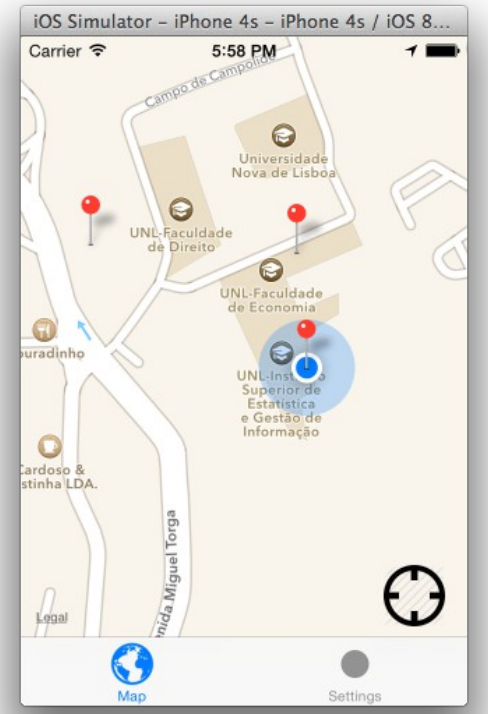

Fig. 23: High location accuracy few seconds after launching the application

[36] http://www.macworld.com/article/1159528/how_iphone_location_works.html [last accessed: 25/01/2015]

Testing can be done with the simulator or the real device, but in the simulator the location is given by the Wi-fi connexion and the simulator cannot track battery consumption.
For these reasons the testing of the applications was done in a real device, in this case an iPhone 4S, and with Wi-fi disabled. The tests lasted 30 minutes and were started with full battery, the same path with the same geofences was followed to minimize the differences between the apps. Due to the GPS limitations the test was done outdoors. Push notifications were enabled in the applications to allow them to run in the background.

4.2.1.- Performance issues

In the native application the push notifications take 2 to 3 seconds to be delivered to the device, and in the external SDK application the notifications take up to two minutes, the reason why this time is longer is because in the native, the notifications are handled locally, and with the external SDK the notifications are handled remotely, the application has to connect with Apple Push Notification service (APNs), and send the unique device token so the APNs can send the notification to the right device[37].

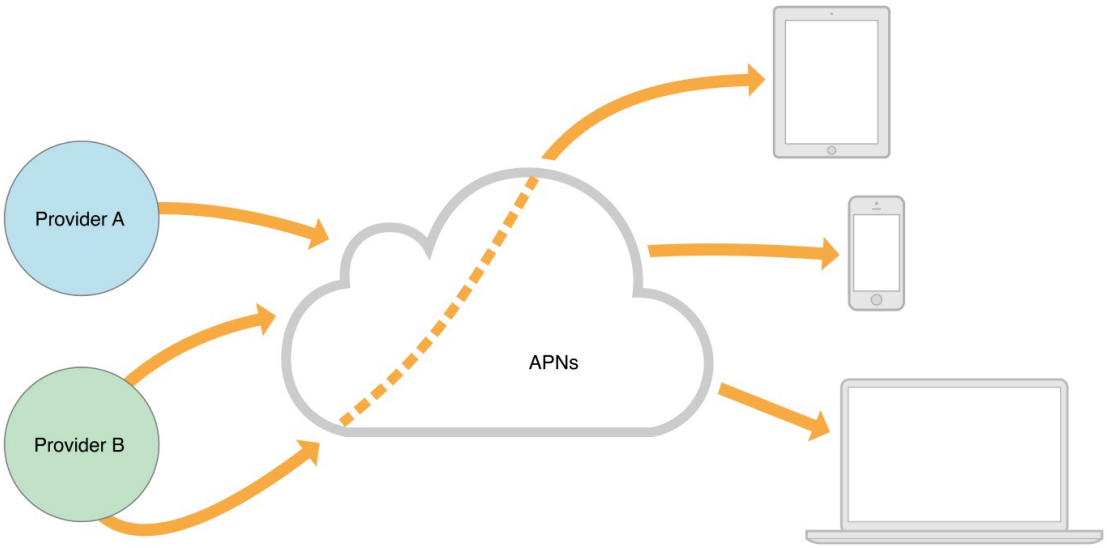

Fig. 24: Pushing remote notifications from provider (apps) to devices

[37] https://developer.apple.com/library/ios/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/Chapters/ApplePushService.html [last accessed: 25/01/2015]

Some of geofences tested were less than 100 meters of diameter and apparently the alert is only sent if the estimated location is inside the geofence, which means that when the alert is sent, the device location is for sure inside the geofence, thus some alerts take longer to be received, because it takes time to have enough accurate location.

4.2.2.- Battery life issues

Battery life was tracked in all the applications and saved in a file together with all the console logs in the following format:

2015-01-19 14:31:22.982 GeofencingTest1[1585:423382] Battery level 1\
2015-01-19 14:31:22.982 GeofencingTest1[1585:423382] new latitude, longitude
+39.9945721, -0.0732978\
2015-01-19 14:31:24.768 GeofencingTest1[1585:423382] new latitude, longitude
2015-01-19 14:32:49.400 GeofencingTest1[1585:423382] Device entered region Residencia\
2015-01-19 14:36:03.455 GeofencingTest1[1585:423382] Device entered region Humanas\
2015-01-19 14:36:33.200 GeofencingTest1[1585:423382] Battery level 0.99\
2015-01-19 14:37:17.419 GeofencingTest1[1585:423382] new latitude, longitude
+39.9956126, -0.0732976\
2015-01-19 14:38:15.066 GeofencingTest1[1585:423382] Battery level 0.98\
2015-01-19 14:40:17.166 GeofencingTest1[1585:423382] Battery level 0.97\
2015-01-19 14:41:47.001 GeofencingTest1[1585:423382] new latitude, longitude
+39.9940461, -0.0737340\
2015-01-19 14:42:19.349 GeofencingTest1[1585:423382] Battery level 0.96\
2015-01-19 14:42:45.796 GeofencingTest1[1585:423382] new latitude, longitude
+39.9934340, -0.0735798\
2015-01-19 14:43:20.659 GeofencingTest1[1585:423382] Battery level 0.95\

Battery starts with level 1 which means fully charged, 0,99 level means battery is at 99%. The log also records every update from the GPS with new latitude, longitude and the values received. And it records also every region the device entered or exit.

With the logs from the four applications we can build a chart to compare the battery consumption. Although the tests were performed in the same conditions they are not fully reliable, due to the fact that having the screen on and other applications are also consuming battery, moreover, when the battery is fully charged, sometimes takes more time to decrease than other times.

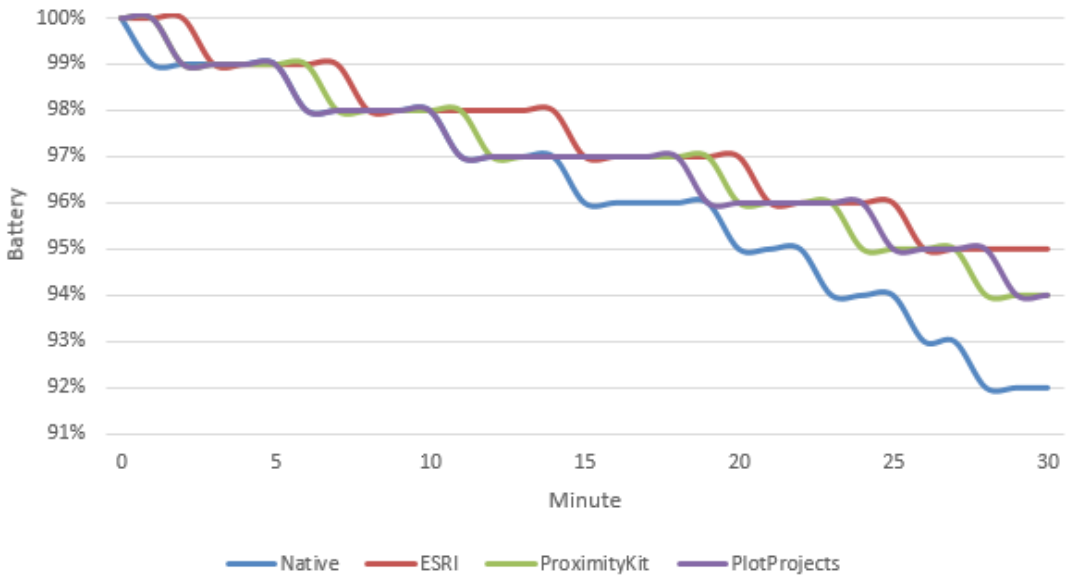

Fig. 25: Battery consumption during 30 minutes of testing.

4.2.3.- Advantages and disadvantages

After the testing, a summary of the tested applications was done, and the decisive characteristic to choose ESRI geotrigger is that it allows polygons as geofences. Also it shows less battery drain and provides API and graphical interface. Another nice advantage from geotrigger is that geofences have tags associated, so users can filter which tags they want to subscribe and only geofences with these tags will be eligible to be sent to the user.

	iOS Native	ESRI geotrigger	Proximity Kit	Plot Projects
Graphic interface		√	√	√
Allow polygons		√		
Lowest battery drain		√		
Provides API		√		√
Can filter by tag		√		
Locally managed notifications	√			
Shows geofencing usage		√	√	√

Table 2: Main advantages of geofencing applications

5.- Conclusions

Geofencing technique can be easily integrated to SmartCampus or SmartCity mobile applications, and it will allow users to receive information from their current location. Data as air quality, temperature, power consumption, the risk of heavy precipitation and even other parameters like high vandalism, or shop commercials and offers could be retrieved and used to inform users with the same interpolation technique used in this dissertation.

Due to the fact that the service is divided into two parts, the data can be provided in sample points and be processed by both services, or provided in a GeoJSON format and be integrated to the geotrigger by the second service. Polygon data from buildings and alerts related to these buildings could be also added to the system.

If many data are added to the system, there could be geofences overlapping, so two or more alerts could be delivered to the device at the same time, however, with the geofences tags combined with the polygon creation there will never exist two geofences overlapping attached with the same tag.

6.- Further Work

All this process could be automatized if the sensor data was retrieved in real time, update the values from the sensors, then recalculate polygons and create geofences from them. But there are not enough fixed sensors tracking air quality information in Castelló, only three and they are located outside the urban area, so not useful for our purpose.
This same process could be applied for more variables in the city or in the university campus, as power consumption, temperature, traffic alerts or building related alerts.

Regarding the air quality data, in our case CO^2 concentration, more accurate information could be provided if we had wind speed, wind direction, temperature, humidity and other factors which affect the real value in the sample locations.

These data are not implemented to be visualized by normal users, they only get notifications, only the administrator can manage the geofences and see the whole set and their properties, thus, to include both a nice way of visualizing geofencing data and a very fresh technology as augmented reality, the service could return a drawable object to visualize it.

These services could also improve the navigation tool inside SmartCampus or even future smart cities, for example, calculating a healthy path not entering in the regions with worse air quality data, or calculating an accessible path for handicapped users without entering not user friendly areas.

7.- Bibliographic references

Bulut, M. F., Yilmaz, Y. S., Demirbas, M. (2013). LineKing : Crowdsourced Line Wait-Time Estimation using Smartphones. *Mobile Computing, Applications, and Services,* pp. 205-224

Espinoza, F., Persson, P., Sandin, A., Nyström, H., Cacciatore, E., Bylund, M. (2001). GeoNotes : Social and Navigational Aspects of Location- Based Information Systems. *Ubicomp 2001: Ubiquitous* Computing, pp. 2-17

Harter, A., Hopper, A. (1994). A Distributed Location System for the Active Office. *IEEE Network*, *8*, pp. 62–70

Ijeh, A. C., Preston, D. S., Imafidon, C. O., Watmon, T. B., Uwaechie, A. O., Cooke, M., Lancaster, P., Widdess, A., Soremekun, M. (2010). Geofencing Security Engineering. *Proceedings of the International MultiConference of Engineers and Computer Scientists*, *II,* pp. 969-974

Junglas, I. A., Watson, R. T. (2008). Location-based services. *Communications of the ACM*, *51*(3), pp. 65-69

Junninen, H., Niska, H., Tuppurainen, K., Ruuskanen, J., Kolehmainen, M. (2004). Methods for imputation of missing values in air quality data sets. *Atmospheric Environment*, *38*, pp. 2895-2907

Küpper, A., Bareth, U., Freese, B. (2011). Geofencing and Background Tracking – The Next Features in LBSs. *INFORMATIK 2011 - Informatik schafft Communities*

Ludford, P. J., Frankowski, D., Reily, K., Wilms, K., Terveen, L. *(2006). Because I Carry My Cell Phone Anyway : Functional Location-Based Reminder Applications. Proceedings of the SIGCHI conference on Human Factors in computing systems,* pp. 889-898 ACM

Paek, J., Kim, J., Govindan, R. (2010). Energy-Efficient Rate-Adaptive GPS-based Positioning for Smartphones Categories and Subject Descriptors. *Proceedings of the 8th international conference on Mobile systems, applications, and services,* pp. 299-314 ACM

Pourhomayoun, M., Jin, Z., Fowler, M. (2012). Spatial Sparsity Based Indoor Localization in Wireless Sensor Network for Assistive Healthcare. *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, pp. 3696-3699

Reed, J. H., Krizman, K. J., Woerner, B. D., Rappaport, T. S. (1998). An overview of the challenges and progress in meeting the E-911 requirement for location service. *IEEE Communications Magazine*, *36*(4), pp. 30-37

Schiller, J., Voisard, A. (2004). *Location-based Services*. San Francisco: Morgan Kaufmann. Retrieved from http://elsevier.com, pp. 1-294

Sheth, A., Seshan, S., Wetherall, D. (2009). *Geo-fencing : Confining Wi-Fi Coverage to Physical Boundaries, Pervasive Computing* pp.274–290

Sluiter, R. (2008). Interpolation methods for climate data: literature review. De Bilt, Royal Netherlands Meteorological Institute (KNMI).
http://www.knmi.nl/bibliotheek/knmipubIR/IR2009-04.pdf

Wong, D. W., Yuan, L., Perlin, S. a. *(2004). Comparison of spatial interpolation methods for the estimation of air quality data. Journal of Exposure Analysis and Environmental Epidemiology, 14, pp. 404-415*

Zeimpekis, V., Giaglis, G. M., Lekakos, G. (2002). A Taxonomy of Indoor and Outdoor Positioning Techniques for Mobile Location Services. *ACM SIGecom Exchanges*, *3*(4), pp.19-27

ZHUANG, Z., KIM, K.H., SINGH, J. P., (2010). Improving Energy Efficiency of Location Sensing on Smartphones. *Proceedings of the 8th international conference on Mobile systems, applications, and services,* pp. 315-330 ACM