Android project proposals

Luca Bedogni, Federico Montori

13 April 2018

Abstract

In this document, we describe three possible projects for the exam of "Laboratorio di applicazioni mobili" course. Each student can choose a project from the set, or suggest something else based on his/her personal interests. In this latter case, project proposals should be submitted via e-mail to Federico Montori (federico.montori2@unibo.it) , with a brief description of the application goals, contents and requirements. The following project descriptions should be considered as hints. Students are strongly encouraged to expand the tracks, adding new features to the applications, and/or further customizing the contents.

1 Wireless Connectivity Map

In the following project the student is required to implement an application that is able to map the wireless connectivity in a geographic area. The goal of the application is to obtain a heathmap of a geographic area in which the strength on the wireless signal (the RSSI) is mapped to a color scale (e.g. red to green). The values represented on the map are to be obtained through actual readings from the wireless network interfaces of the smartphone. This has to be performed for the following technologies: UMTS (3G), LTE (4G) and WiFi. More in particular the requirements of the application are as follows:

1.1 Use of the Google Maps APIs

The application has to map the position of the user as well as any data collected in the geographic map. As the user moves in the map (when the application is active) a value has to be produced by the application and it has to be reported on the map in the form of a color associated with the area the user is in. Google Maps uses the GPS coordinates to encode locations, however, the student has to shift to a location (point) based encoding to an area encoding. The student can choose the way he/she wants to encode areas, with the only constraint that the whole map can be covered (e.g. if we use non overlapping circular areas there will be unavoidably "holes" in the map). hint: one way to do it could be using GPS-dependent square areas, or MGRS coordinates, or overlapping circular areas.

1.2 Convert the RSSI in a color encoding

The received signal strength in every area should be reported on the map in the form of a clear color scale. Whenever the user occurs within a new area, such area has to be mapped and reported on the map. Whenever a user occurs within an already mapped area in a reasonably distant point in time (e.g. after 1 day, or it can be asked as a configuration parameter in the app) a new value has to be generated and the mean of the value readings in the same area should output a new color. The number of the reading for each area should be taken into account too (e.g. using more transparency for areas with less readings).

1.3 Three technologies

The application should be able to switch among three different maps depending on the technology being used, in other words, each technology (UMTS, LTE, WiFi) outputs a different map.

2 App Monitor

In the following project the student is required to implement an interactive application able to monitor the applications in background and sending an usage report. The project has to fulfill the following features:

2.1 Monitor the Apps in background

The application need to be able to monitor the applications in background and, in particular, it has to provide at least three global metrics (e.g. how many apps are running, for how long, the mean within the last few hours etc.). Furthermore, the application has to provide at least three metric for EACH application in background (e.g. how long has it been running, etc.) Obviously such metrics can be expanded and each student is encouraged to do so.

2.2 Interactive selection

The user has to be displayed with a menu with which settings for the application can be adjusted, in particular:

- Setting about which apps to monitor, which metrics to use etc (for instance, a user might not want to monitor social apps and has to indicate which ones have to be excluded, whereas others might not be interested in the total time, but only in a time frame etc.).
- Settings about how and where to send the report (some users may not want to receive daily report, but only one per week, others may want to be updated once every 6 hours etc.).

2.3 Sending the Report

The report has to be sent either periodically or on demand (according to the applications settings). Options can be via e-mail, via instant messaging or as a post on a page. Settings have also to include the periodicity of the report. A custom server is also a possibility, although its implementation is not evaluated as part of the exam.

3 Participatory Crowdsensing App for Citizen Science

Participatory Crowdsensing is a paradigm for which mobile users (participants) perform tasks issued by a crowdsensing campaign owner in order to collectively monitor phenomena of common interest (e.g. taking pictures of road holes, monitor the noise pollution in an urban area through sharing sensor readings, a concrete example is given by the road accident observation feature in Waze). The advantage of such applications are in the low cost: an entity interested in monitoring a phenomenon does not have to deploy sensors in an area, nor hire specialized team to report observations, rather it aggregates observations provided by end users who would participate in such campaign. The student is required to implement a client for a participatory crowdsensing application, which should be able to:

- Notify the user of new incoming tasks and accept/reject them (taks are issued as twitter posts and results are sent back as replies to the post).
- Notify the user whenever the conditions for accomplishing a task are met (e.g. the user is in the area of interest at the desired time).
- Perform interactively the task (each task is either taking a picture, perform a sensor reading or record an audio) and send the results back.

3.1 The task and its format

Each task is a JSON object with the following format:

```
{
  "ID" : "LookForKoalas"
  "issuer" : "LAM_UNIBO_2018",
  "type" : "picture",
  "lat" : "-37.835309",
  "lon" : "145.047363",
  "radius" : "1.0",
  "duration" : "5",
  "what" : "eucalyptus_trees"
}
```

- **ID** [string] is an arbitrary string for uniquely distinguish the task.
- **issuer** [string] the Twitter account of the issuer.
- **type** [string] it's either "picture" (the user has to take a picture of the object), "light" (the user has to use the light sensor), "noise" (the user has to record the dB value of the microphone), "temperature" (the user has to record the temperature value in Celsius) and "RSSI" (the user has to send the signal strength of the cellular technology).
- **lat** [float] Latitude of the task center
- **lon** [float] Longitude of the task center

- **radius** [float] radius (in meters) of the task (thus, the task can be performed within a circular area)
- **duration** [int] How long is the task valid since it has been posted (in days).
- **what** [string] what are we interested in, this is an arbitrary string, it only contains information on what we are aiming to observe.

3.2 Receiving the task and sending back the results

The tasks are issued as twitter posts, containing the hashtag #LAM_CROWD18 and the JSON string, thus the twitter API has to be used. The application has to be notified whenever a new task is being posted (a filter on the twitter account may be used) and the user has to manually accept it (or reject it). A twitter account (@LAM_UNIBO_2018) has been set up as "the server" and it will periodically post tasks, however students can create their own, or simply post tasks on their twitter page. The ID of the post has to be saved, since replies to such task are to be posted as post replies (containing the data). No format is required for the response.

3.3 Performing the task

Once the task has been accepted it has to be saved locally until it is performed. Whenever the conditions for performing the task are met (the user is in the correct area within the task duration) the user has to be notified about the chance of performing the task. Then the has has to be performed interactively (e.g. the camera should be turned on within the app and the user can take a photo). When the task is performed, the app has to send the results back as a post reply and the task is deleted from the local "list".