Bulletin of the *Transilvania* University of Braşov • Vol. 8 (57) No. 2 - 2015
Series I: Engineering Sciences

THE DEVELOPMENT OF AN INFORMATION SYSTEM FOR TOURISTS USING THE ANDROID PLATFORM (II)

S.A. MORARU[1] A.C. MANEA[1]
D. KRISTALY[1] C.L. CRISTOIU[1]

Abstract: *The interest for mobile applications has grown increasingly in the recent years and this thanks to the easiness with which their users can be informed in a very short time. The mobile application to inform tourists is used on the smart mobile terminals presenting the sights of a city, providing information about each point of interest and information on how the user can reach it. This paper presents a system that is addressed, primarily, to tourists coming to visit the city of Brasov, but it can be used equally well by the inhabitants of Brasov who have not yet discovered all the attractions of the city. From technical view, this paper tackles the development of the client application.*

Key words: *mobile application, file, android, interface, profile.*

1. The Development of the Client Application

1.1. The Structure in Android

The *src* folder contains packages which in turn are composed of Java classes.

The *res* folder contains in turn other important folders:

- *anim* - contains XML files that perform certain animations on the user interface;
- the images used are saved in the folder *drawable*;
- the XML files that compose the windows of the application are located in the *layouts*, while the menus pertain to the *menu* folder*;*
- the names of buttons, the hints and the names of the menus are declared in the *string.xml* file from the *values* folder. Here the *strings* for each language are also defined, the application being available in English and Romanian.

The *Manifest.xml* file contains:

- the name of the package where the application is;
- the version of the application;
- the requirements for OpenGL ES 2.0. necessary for the use of Google Maps;
- the declaration of all the permissions the application requires when installing it (accessing the approximate location based on the network or the exact location based on the network and GPS, modifying or deleting the content of the SD card, deleting the content of the SD card, reading the Google service configuration, full access to the network, connection and disconnection from the Wi-Fi, seeing network connections, running services on

[1] Dept. of Automation and Information Technology, *Transilvania* University of Braşov.

start-up, testing the access to the protected storage);

- the maximum and minimum version of sdk, which influences its minimal version of Android on which the application can run;
- specifications regarding the dimension and density of the screen;
- declaring all activities and services;
- metadata that specify what API_KEY was used to access GoogleMaps [6], [11].

Besides the main project *Tourist*, certain bookstores were used such as:

- *the library* - used to achieve a compatibility between the *ActionBar* widget and *Fragment* with the devices running with an operating system version 3.0 less than the *3.0 Honeycomb* API 11 version;
- *Library3* - route used to decode the route received as string by parsing a JSON;
- *Map* - used in order to plot the route on Google Maps [11].

The *Libs* folder contains *JAR* files, Java archives allowing the import of certain classes in order to be used in the project (Figure 1).

1.2. The Interfaces with the Users (the Screens)

The graphical interface with the user (UI) is represented by what is seen on the mobile device at a certain time.

The application client comprises a total of 14 screens: 7 activities and 7 fragments.

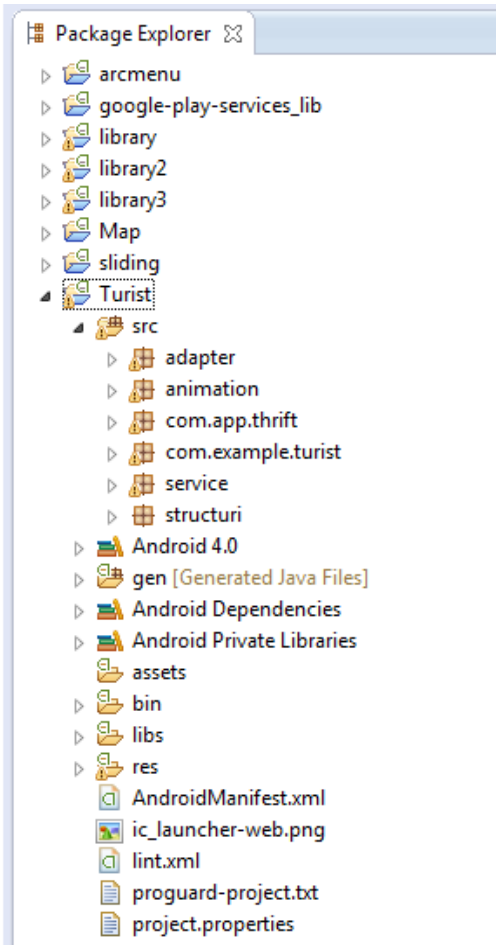

Fig. 1. *The Structure in Android*

Besides the activities and fragments on UI, alert dialogues can also be noted.

The graphical Interface of the application is divided into 3 levels:

1. On the first level there are three activities: *MainActivity, Recover, Register;*

2. On the second level there is only one activity *ContextAfterLogin* managing the three fragments: *LocationList, FriendsList, Profiles;*

3. On the third level there are two activities: *Location* and *Chat*. *Location* activity is responsible for managing the work of four fragments*: InfoLocation, GalleryPhoto, BusList, ReviewList.*

1.3. The User Login Window

The user login window represents the actual main activity which is executed first, with the launching of the application. The source file is represented by the class *MainActivity.java*, a class where all the components of the graphic interface are initialized and in which the events occurring on the UI are managed [5].

The source file of the interface is represented by the *mainactivity.xml*. In order for the UI to come to life, a *ScrollView* was used, a widget that acts as a window with vertical navigation bars when the screen of the mobile device on which the application is run has a smaller diagonal, or when the application runs in the landscape mode.

The header of the window in which the name of the application appears is framed in a LinearLayout. In order to collect the login information, two EditText widgets are used that have set a certain hint in order to indicate the user how the respective field must be completed [4].

Also in the same window, the user has the option to open a new window to create an account, another window to recover the password and a third window for details about the usefulness of this application after creating the account. In order to set up the three links to the three windows, three TextView widgets are used over which the *setOnClickListener* event is applied, obtained by implementing the *OnClickListener* interface from the *android.view. package*. These widgets as well as the login button are grouped using a LinearLayout, this behaving as a panel having vertical orientation to rank the components one above the other.

Connecting the user to the application is done based on a password and an email address, this being unique in the database. Anyone who has a device with an Android operation system starting from the version *2.3.4 Gingerbread*, can create an account and use this application [8].

The password recovery is based on the email address, subsequently a mail being sent to that address with the password chosen when creating the account. About Section presents the user a brief description of the application, as well as its operating policies. The access in the account can only be achieved when the user has filled both fields and only if there is an internet connection. If these two things are not fulfilled, the user will be warned by displaying on the screen a Toast with the appropriate message.

When launched, the application verifies in the background if the application *GooglePlayServices* is installed on the device, application required in order to use Google services [10]. If the system does not detect the application, then the user is automatically redirected to *GooglePlayMarket* specifically on the page of the GooglePlayServices application, thus avoiding the user frustration induced by the search. If the system detects that *GooglePlayServices* is installed and the mobile device has an active data connection, then it completes the user's connection and passes to the *ContextAfterLogin* activity.

1.4. ContextAfterLogin Activity

ContextAfterLogin activity is the activity that lies behind the three fragments: Locations, Friends, Profile. In this activity, a configuration panel is implemented, comprising the following Sections: **I.** *Edit profile* which gives the user the opportunity to change the name, password and email address associated to the account; **II.** *Conversations* in which the user will be able to track the number of new opinions of the users about a certain sight, but also the number of new messages received; **III.** *The map* allows viewing the position on the map of all the friends who have the Visible on the map option active, but also all the sights; **IV.** *My Account* contains the user's disconnect option. When accessing the disconnect option, an alert dialogue appears (*AlertDialog*) that warns the user that if he quits the application, he will no longer have access to its services.

Default by disconnection, the user becomes invisible on the map even if when accepting disconnection, he was visible.

The xml source file of the interface with the user consists of a *RelativeLayout* that has the following components:

- A *DrawerLayout* used to implement the configuration panel;
- A *ViewPager* implemented for navigating among the three fragments mentioned above. This in turn contains a *PagerTabStrip* used to implement the names of the fragments on the UI, but also to customize the tabs or the indicator of the currently selected tab;
- *A FrameLayout* represents the actual content of the three fragments.

When the user accesses one of options from Editing profile in the background it is running a class that derives from class *AsyncTask* allowing the display on the user's interface of an AlertDialog through which the user can change his/her username, password or email address.

These changes do nothing but send an update request to the server and this forwards it to the database [3].

After finishing the update, the server will bring the new data from the database. As the configuration panel is accessible from any screen of the application from the second level of the screen architecture, so the user can disconnect both from the profile view, the screen with the list of friends as well as the screen with the list of sights.

The application also offers a screen containing Google Maps, map on which the markers with all the friends visible on the map are placed, but also all the sights.

With a simple touch on the screen of a marker, a route between the user's marker and the respective marker is drawn on the map. In addition to the route drawn, a window containing the distance in meters between the two points also appears [9]. The sights are identified by pink-violet markers, the friends' markers are orange while for the current user connected, the option was for blue. In order to help the users, the map offers support for rotation events, zoom in and zoom out, recognizing the gestures made on the screen by them, but also using a controller available on the screen. Besides all that appears on the screen, a button used to return on the map to the user's position is also to be found.

The menu comprises four options: *Walking, Biking, Driving* and *Clear route.* The first three options can be used to change the route depending on the means of transport chosen, the route drawn on the map for walking *(Walking) being different from that in the case of a means of transport (Driving).Clear route* is used to delete the route drawn on the map.

Also at the top of the screen there is a drop-down list from which the user can select the type of visualization of the map: Hybrid, Satelyte, Normal, Terrain, None. Switching between the 3 fragments above referred is achieved by implementing the

ViewPager widget and for the user to feel a feedback from the application, on the transition from one fragment to another, an animation appears which uses the basic style in order to display the fragment on the left, while on the right side one, a depth effect is applied (fade-in/fade-out).

The location screen. The main screen contains a list containing of the tourist sights. The section for each location provides general information, a photo gallery and a map with the bus routes running near the location, but also a list with the opinions of the users who have already visited that location.

In the newer design versions of the user interfaces, the simple lists as *ListView* or *ExpandableListView* were successfully replaced by *GridView* which is actually a list built on a table [12]. This widget presents in a more convincing manner the content.

The list of sights presented to the user is built on this *GridView and is personalized with an adapter ImageAdapter.java, a class extending the BaseAdapter from the android.widget.package.* The list presents the users a minimum of information about the respective locations: a specific image and its name.

In order to make sure that the list occupies the whole screen available to the mobile device both the width and the height are set as *match_parent* (whatever diagonal the mobile device has, as well as its orientation (portrait or landscape), the list will always occupy the whole screen) [15]. For a more favourable framing, the number of columns is chosen to be three.

A faster Browsing of the list can be made using a scroll bar:

fastScrollAlwaysVisible = "true".

In order to help users, the application has a menu implemented that allows them to filter the sights according to their type: churches, museums, towers, libraries and others, also having an option that allows returning to the initial list with all the sights [16].

In order to send feedback to the user when he touches an item from the list, the object named *Gridview* selects the method *onItemClickListener* [1], [12].

Activity Location. This activity implements the ActionBar concept. Using the tabs together with the fragments is similar to *ViewPager* in what concerns the navigation between screens, reducing the number through which the user should have surfed.

Equally important is the navigation back to the previous activity; this is done either by using the *Back* button whose navigation is based on browsing the history of the screens which the user surfed in reverse chronological order or using the *Home* button that surfs a level above the UI architecture [14].

Whether the user is on the first level of the screen architecture, whether he is on the third level, he has access to the *UserMap* screen and the button on the *ActionBar* on the right, he may easily follow at any time his position in regard to the sights.

The information screen. Within this screen, the users can read a short history about the respective sight.

Presenting a brief history of it can quickly capture the interest to visit. Many people are interested in the history of the place where they go, but also by what they can see inside the sight or its architecture, this aspect being visible in the photo gallery.

Besides this brief history, the application makes available a program to visit the sight, an essential aspect concerning the sights requiring an entry permit as well as the phone number to which they can call in order to ask for details

Each user can share the preference to go to a particular sight through a voting system of the application. With its help, a statistic of the people who want to visit the respective sight is maintained.

The users who have already visited a sight can vote the fact that they were there. Following this voting system, a statistic is obtained representing the number of people who were visiting a certain sight, as well as the number of people who want to go to visit the sight in question. The purpose of this voting system is to help tourists decide more easily in respect of visiting a particular sight because of the simple fact that today some people's decisions greatly influence others' decisions.

Either out of curiosity or by mistake, a particular user can vote without being really in that location or without really wanting to go in that location. In this case, users have a cancellation option of their vote.

The Photo Gallery Screen. The Android application offers a gallery of photos that contains a minimum of pictures as representative as possible in order to attract tourists towards the sights [2]. The interface is implemented based on an *ImageSwitcher,* a widget that allows users to view the gallery by simply touching the screen from right to left or left to right at the bottom of the picture. Besides *ImageSwitcher,* a *Gallery* widget is used that acts as a frame which allows an increased viewing of the images.

Buses screen. The main element of the application is to inform the tourist when accessing this screen which means of transport is needed in order to reach the targeted sight. The graphic interface contains an *ExpandableListView* [7] which contains details about the route, details like: the name of the departure station, the name of the terminal station, the time the means of transport leaves as well as when it reaches the station near the sight, the number of stations in which the means of transport will stop until the terminal station, but also the length of the route in kilometres. The first thing that the application does behind what the user sees is to identify depending on its coordinates the street he is in and the street where s/he should arrive. This is done based on a specific Google service, namely *Reverse Geocoding* based on an http request:

Google Directions API is another service used to obtain the actual details about the route. Unlike *Geocoding API* which uses the geographic coordinates, *Google Directions API* receives as a starting point the street the user is in, transmitted o*n Reverse Geocoding,* and as a final point, the street where the sight is, obtained in the same way, based on its latitude and longitude. In order for the route identified to be the right one also when the user accesses the application, the time taken directly from the mobile device is also forwarded to *json*.

The next step is to identify the nearest public transport station, where the user will take the bus needed. In order to help the user even more, besides the bus route, the route to the bus station is also drawn, but also the route from the arrival station to the sight.

In order to facilitate the user's orientation on Google Maps [16], the position where s/he lies is identified by a circle, and a *marker* is placed for the terminal station as well as for the sight.

As noted above on the Information screen, the others' opinions matter a lot in the decisions that other people take. Following this principle, as well as the voting system *Information screen*, it was decided to implement a screen containing other people's opinions, users of this application, about the sights they have visited [14]. Each post is identified with the picture of the person who posted his/ her name, the message itself and the date and time when it was sent.

Friends Screen. Although the application is based on location and informing the users about the sights of Braşov, it has implemented a part of communication with other users, added as friends [16]. The implementation maintains the same

principle, the use of *GridView* widget, which is more significant than using a normal list. When a user touches a user's picture, a widget is displayed on the screen that contains three options: *Send message, About and Delete.* The idea of communication has a similar role to the voting system on the Information screen, as well as that of the *Personal Opinions screen,* namely to inform the user. The discussion with a friend who has been to visit a sight can greatly influence the decision to go there or not.

What is more important and relates to the possibility of seeing friends on the map is the fact that when the tourist is undecided to go there, he can check in his friends list who is online and then can see on the map by the option available in the configuration panel, if any of the visible friends are visiting the sight that he intended to go to.

The communication chat is also represented through an Android activity that starts when the user accesses the option *send message*. The difference between this activity and the ones occupying the entire screen is made by its different declaration within the *Manifest.xml file*, by applying a different theme. Setting the theme of an activity as being the type *DialogWhenLarge* allows the launching of the activity in the form of a dialogue [13]. The updating of the messages list is done using a *Broadcast Receiver*. Besides the possibility of communication, the user can see a route directly from him to a certain friend if the latter is visible on the map, which is accessible from *UserMap.*

The second option, *About,* opens a dialogue with details of the user that was accessed: his picture, the street he is on or was the last time he was available on the map, his status, the sights where he was and those where he would like to go.

The third option is obvious, allowing to delete a certain friend, with the possibility to add him later in the menu.

The menu allows filtering the list of friends namely: friends visible on the map, an option that brings the list to the initial state namely with all the friends regardless whether they are visible or not on the map. Also from the menu, the user can add other users as friends.

The screen profile is divided into three sections: a section is represented by the user's personal details, the second section contains a mini statistics which specifies the total number of friends, the number of friends online in the communication chat, as well as the number of friends visible on the map at a time, and a third section is represented by a list in which different routes of certain means of transport are updated. The list will be updated with a service that will continually check the user's position and will verify which of the 36 sights is closer to him (maximum 5 sights). This will contain the image of the sight, its name.

The section of personal details gives the user extremely important information, namely its location, respectively the street. In addition, the application offers the user the freedom to decide whether he wants or not that the others friends could see his position on the map by the option "Visible on the map". Also, the users can change their profile photo.

With a simple touch on the current image, the application sends you in the photo gallery of the mobile device. The menu on this screen consists of two options: *About and Delete account. About option* is similar to the *option About* found on the login screen, giving a brief description of the application.

The second option, *Delete account* can be used if the user wants to delete the account permanently, this being forced to create a new account in order to use the application. To delete the account involves deleting all the personal data, as well as losing the friends list.

2. Conclusions

The application developed can be of real help to people who want to visit the city of Brasov and the facilities offered are able to provide valuable information on the optimal routes that a tourist can follow to arrive soon and safely to the targeted sights.

The client application has been optimized to run on different resolutions. Also, it can be used on mobile devices having different screen diagonal. The user can use the application both in the portrait and landscape mode. It can run on both phones and tablets but the minimum version of Android being 2.3.4 on tablets.

Regarding the private location service, the users are not limited to the mobile devices with GPS, the location being made based both on the GPS (exact location) but also based on the network (approximate location).

With the help of the *Text To Speech Service*, the users can control vocally the application, surfing between its screens.

References

1. Gargenta, M.: *Learning Android.* O'Reilly. 2011.
2. Lauren, D., Shane, C.: *Android Application Development Second Edition.* In: SAMS Publishing, 2012.
3. Meier, R.: *Professional Android 2 Application Development.* Wrox Professional.
4. Meier, R.: *Professional Android™ Application Development*. Wiley Publishing. 2009.
5. *** 2.5 Creative Commons Attribution: *Getting Started | Android Developers*. Available: http://developer.android.com/training/index.html. Accessed: 26.08.15.
6. *** Android-er: *A simple example using Google Maps Android API v2*. Available: http://android-er.blogspot.ro/2012/12/a-simple-example-using-google-maps.html. Accessed: 06.08.2015.
7. *** Android Asset Studio. 2010. Availabel: http://romannurik.github.io/AndroidAssetStudio. Accessed: 09.09. 2015.
8. *** Android App Patterns. Available: http://www.android-app-patterns.com/. Accessed: 09.09.2015.
9. *** Brian, F.A.: *Thrift for Windows and MSVC 2010*. Available: http://www.brianfosterallen.com/thrift-for-windows-and-msvc-2010/. Accessed: 16.08.2015.
10. *** Developers, Google: *Google Maps Android API v2 - Google Developers*. Available: https://developers.google.com/maps/documentation/android/start?hl=ro#install_and_configure_the_google_play_services_sdk. Accessed: 05.08.2015.
11. *** Developers, Google: *The Google Directions API - Google Maps API Web Services - Google Developers*. Available: https://developers.google.com/maps/documentation/directions/#TravelModes. Accessed: 09.09.2015.
12. *** Mayani, P.: *Android - GridView example*. Available: http://www.technotalkative.com/android-gridview-example/. Accessed: 06.08.2015.
13. *** McKenzie, D.: *Designing for Android|Smashing Magazine*. Available: http://www.smashingmagazine.com/2011/06/30/designing-for-android/. Accessed: 09.09.2015.
14. *** *Open Database Connectivity - 'ODBC'*. Available: http://www.anaesthetist.com/mnm/sql/odbc.htm. Accessed: 06.08.2015.
15. *** Tamada, R.: *Android Sliding Menu using Navigation Drawer*. Available: http://www.androidhive.info/2013/11/android-sliding-menu-using-navigation-drawer/. Accessed: 09.09.2015.
16. *** Teta Infoimpex SRL. Braşov. *Obiective turistice*. Available: http://www.welcometoromania.ro/Brasov/Brasov_Lista_Obiective_r.htm. Accessed: 05.08. 2015.