Android Code Camp 2010

LunchMagnet Project Report

Group 16
Harri Johansson
Antti Knutas
Tommi Kähkönen

1. Introduction

As a part of the Android Code camp the software team implemented an online service called the LunchMagnet. The idea is simple: Even today, when a great part of the population has smartphones, it is not a trivial task to discover who of your friends are in town. There are a lot of other use cases as well: You can use the mobile service to find out which of your friends are in the same mall as you are and would like to go shopping, and so on. The service consists of two parts: A mobile software program that runs on the Google Android platform, and a Java Web Service whose network interface has been implemented as a servlet.

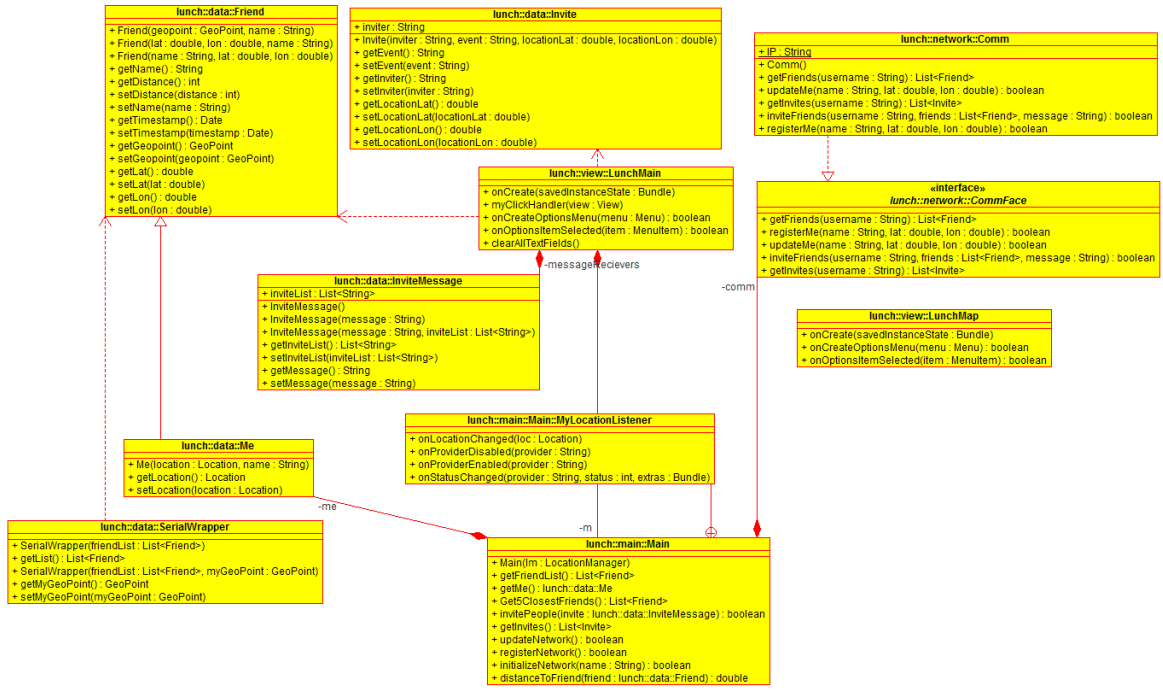
In the following chapters first the technical implementation details are explained, after which there is a short chapter on possible use cases of the program and a short chapter on the development tools used. The technical implementation explains the details of GUI implementation, data structures and some finer details where necessary.

2. Technical implementation

LunchMagnet has been programmed using Google's flavor of Java like most if not all Android programs. It consists of three main classes and two activities: The so called main class which contains the program controller and contains the models, the two view, or Activity classes, and a separate class that handles networking. The separation has been done in the spirit of Model-View-Controller best practises, though some shortcuts

had to be taken because of short development time.

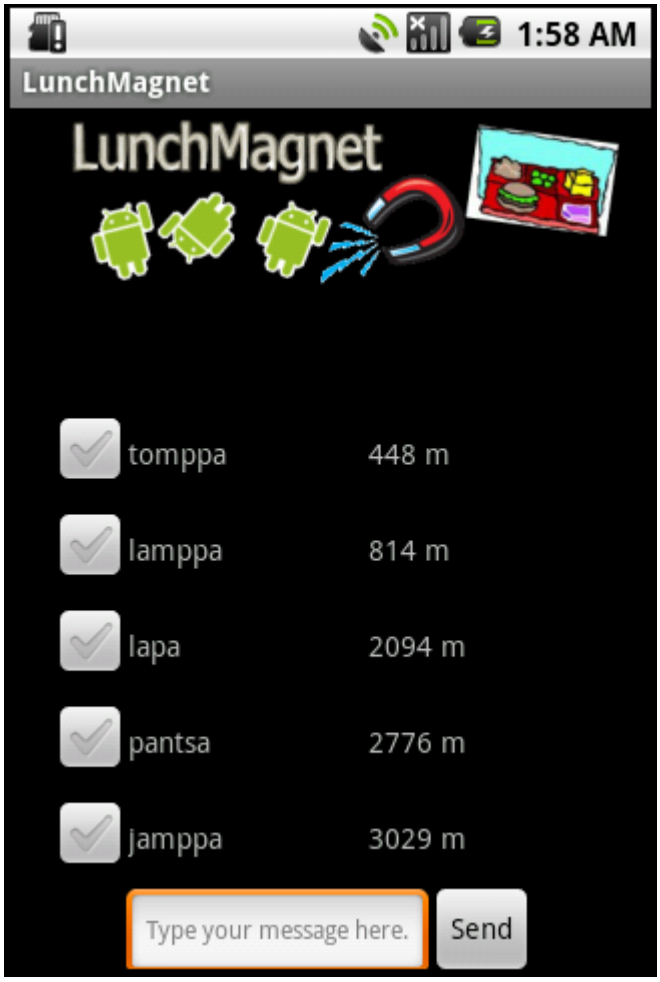
The class diagram is visible in the picture one below. As one an see, the main class contains most of the main logic and the main datatypes. While they are not displayed, the friends are stored as a List, and Me is a simple variable.



Picture 1. Class Diagram

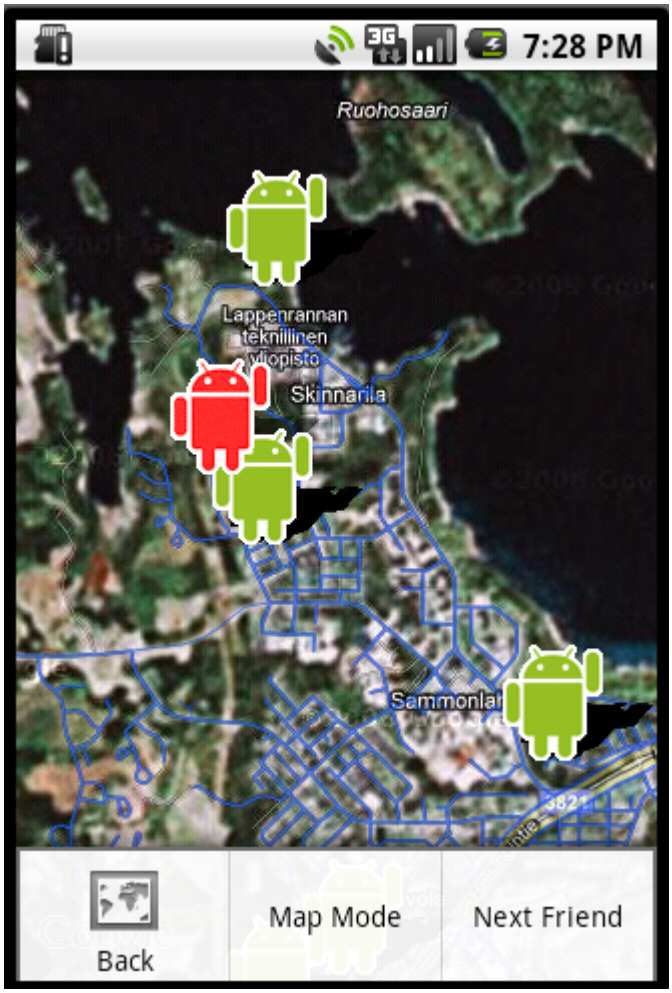
2.1 Graphical Interface

Two different activities are used in the graphical user interface of *LunchMagnet*. When application is run, the main activity is loaded. After user has typed his name to the corresponding field and clicked *register* button, the UI will call the method of main object that asks the server to send information about user's closest friends to the UI. This information displayed in a *tableview* that contains a *checkBox*, friend's name (*textField*) and distance (*textField*) in one row. At the moment, only 5 closest friends can be displayed on this view but as the application is developed further, some dynamic structure will be utilized. Below the *tableView* there is a button that sends the message (typed in *inputText* -field) to the server. User can select the receivers of the message by selecting a corresponding *checkBox* right next to the friend's name.



Picture 2. LunchMagnet Login screen

The android menu found in android.menu library is utilized in the application. By clicking the menu button in a phone, a menu will appear to the bottom of the main screen. The menu has options for launching the *mapView*, updating the distances of user's closest friends and viewing the invitations that the user has received from others. By clicking *show invitations* button will hide the checkboxes and display the friends' names and messages instead of their locations. *Update friends* button will update the friend list and get new distances from the server. If the user clicks the *show map* button, the application will start another activity and launch a *mapview* that displays the google map. User itself is displayed as a red marker and all his friends are displayed as green markers. If the user clicks phone's menu button, this will bring the menu on the screen. Menu gives options for changing the map mode, locating the next friend (the map is centered to the next friend) and going back to the main menu.



Picture 3. Map View

The information from the *mainActivity* (user's location and the list of his friends' locations) transfered to the *mapActivity* by using the *bundle* -functionality. This will probably be changed in the future. By using the singleton -design pattern, the mapView would be able to access the information it needs directly.

The user interface needs to be developed further, because some very basic things are missing at the moment. For example, some pop-up messages should be used to tell the user what is happening: if message is successfully sent or if an invitation is arrived. The list in the main menu should be replaced by a *AdapterView* to fill the layout with data and handle the friend selection. Also, all the elements of the UI should be named so that the user gets the idea, what kind of information is represented on the screen.

2.3 Network Layer and Web Services

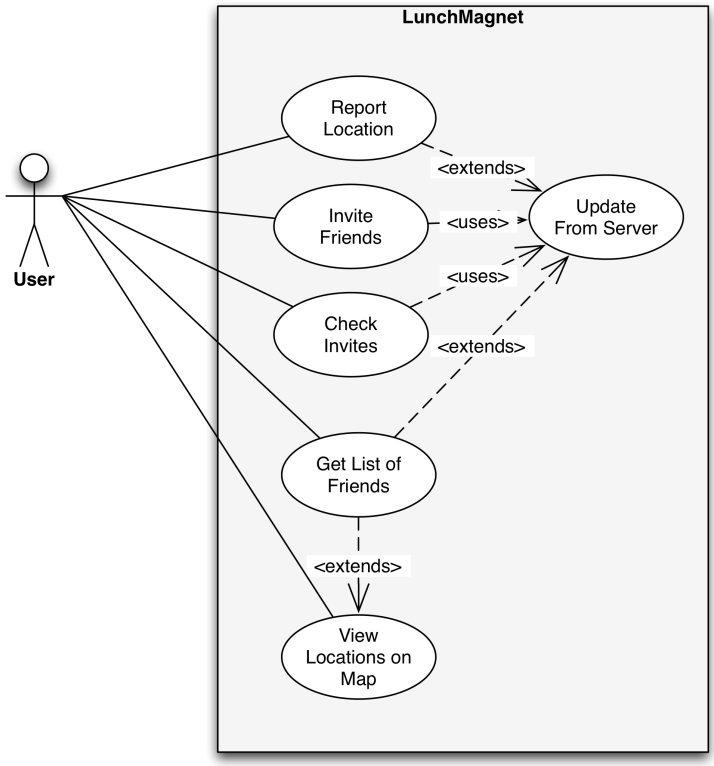
The network layer has been implemented in TCP/IP communications over HTML, in the spirit of RESTful web services though true implementation was not achieved in this time. The main point where the web services implementation differs that data objects are not

accessible through URLs but must be always requested from the same servlet addresses. Otherwise it is assumed that the connection is stateless, and client state is not changing during the data request from the web service.

The actual data request is transmitted in HTTP post to a servlet, which then processes the parameters and replies the data. The data format being used in replying data is not a real standard either, but individual data objects are separated by newlines, and data fields by semicolon. The advantage of this approach is a compactness of data, but requires the knowledge of the format. There is no real metadata available from the server.

3. Use Cases

LunchMagnet has only one planned user group: Ordinary mobile phone users. The administrator functions and other things are really not planned to be exposed to a casual user. Similarly, since it is a mobile phone app, there is a limited amount of use cases. The use cases that are available are detailed in the following picture four.



Picture 4. LunchMagnet Use Cases

4. Development Methods

One of the key factors that made this project successful was well defined work distribution. This means that every member of the team had his own area to focus at. Antti focused mainly at Java Web service, Harri had location service as his primary target and Tommi concentrated on the UI and Google Maps. Of course, these responsibilities were overlapped, especially during the dead line-style coding phase in the end of the code camp. Not all the members could have had enough time to study all the issues of Android-platform and Java Web Services by himself so work distribution made things much easier and saved time.

As a primary development tools, Eclipse with Android SDK (Software Development Kit) and Google Maps API (Application Programming Interface) installed were utilized. The SVN (Subversion) management system was installed as a plugin for Eclipse and the SVN server was hosted by Antti. The SVN offered many benefits and made the development process very agile by allowing rapid changes by all the members of the team. This also meant no more playing with USB (Universal Serial Bus) -memory sticks and breaking the code. The development process involved also considerably discussion and interaction among the team members. This also promoted the XP (Extreme Programming) -style development. The code is a bit clumsy at some places. This is because of very limited time and the need for getting out functional pieces of code very fast. The style of coding is similar to XP: the test code were often written before the actual code to ensure the functionality. All the testing were done with Android emulator - no real phone was used.

For documentation tasks of the code camp, Google Wave and Google Docs -tools were used. The team was not able to work all the time in one centralized place so some remote work had to be done and these tools are very suitable for that purpose. The amount of project planning was rather small because of restricted time. For its part, this caused some problems that could have been prevented by considering some things beforehand. The team realizes the importance of planning, but it is self-evident that in this kind of course, the focus is on coding. However, the used methods and the tools were suitable for this project and the result was very successful.

5. Summary

There were a great amount of code camp spirit among the software team. The used tools and techniques were very suitable for the project but all the members also learned a lot and identify those things that could be done better in future projects. The software team reached the goal that was set during the first day of the code camp and the result of this is a fully functional Android application. LunchMagnet uses readily available software tools and platforms to satisfy a user need that has existed for a long time but is not yet quite fulfilled. LunchMagnet is intended eventually to be developed into a real software service, but is hardly the only one existing right now.