# Introduction

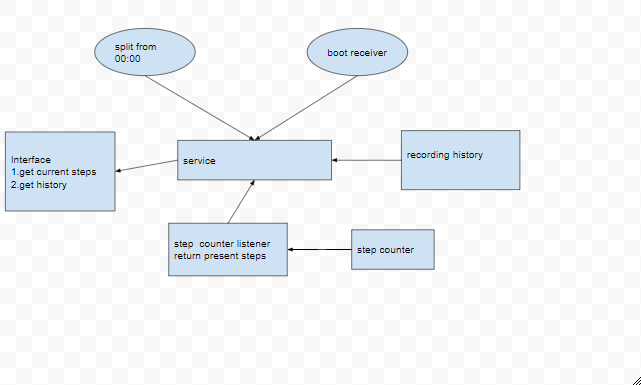
With the development of sensors, smartphones are an ideal tool for monitoring and promoting physical exercise (Harries Tim, 2016). Smartphones are becoming the sensor hubs in a way, opening a rich experience for users. There are multiple different types of sensors which are built in android, such as motion sensors, position sensors and environment sensors (Google, 2018).

There are different ways to calculate the number of steps taken by the user. The first method is based on accelerometer using the Pythagorean theorem to calculate the magnitude of the acceleration vector of each sample. In this way, a service has to keep the device awake for calculating the steps (Martin Mladenov, 2009 ). In other words, the program can prevent the device from going into sleep during a lifetime of your service. But this approach will drain the battery fast. Another different way is built-in step counter and step detector sensor, which internally uses an accelerometer, but Android still treats them as logically separate sensors. Meanwhile, sensors are high battery optimized and consume very low power (Nagpa, 2016). Its high accuracy has been verified.

The report will discuss how to change the existing code from accelerometer to built-in step. Now, there are two different choices. Firstly, the outdated code can be refactored from accelerometer to built-in step counter. Another choice, the existing code can be abolished, the new app will be rewritten.

# methodology

# Results



The flowchart provides an information about the idea of new structure based on the method of step counter. But some functions are omitted such as debug logger and traffic logger. In comparison with the existing application, the main idea is similar. Both of them separate the views, the main algorithms, and services. But I would like to build a module for the Pedometer module and provide an interface, which is easier to test and maintain it.

For the functional aspect, the existing code is more comprehensive than I thought. But I reckon the combination of built-in step counter and accelerometer. In other words, the existing code can add codes for selecting suitable methods based on the edition of Android. It will make app suitable for most Android edition. It is easy to say that we keep the existing arithmetic and add the step counter API, the new application should be suitable for most Android edition. A number of functions are one of the advantages of outdated version, which consider the most functions in research.

According to Harries and Eslambolchilar (2016), the implemented functions have been listed below:

1. There is no requirement for additional equipment such as pedometers and no need for data entry.

2. The application counts steps continually and automatically.

3. the application was disabled until the start of the trail when it was remotely enabled

In contrast with the function in the source code, it is just the tip of the iceberg. The elder version considered about researching the effect of physical activity, in which the participants were divided into three groups: a control, an individual feedback groups and a social feedback groups. The application has considered about groping.

For the version of the application and supported package, the elder edition was developed in 2012, and the MinSdkVersion was 7. The author didn’t write down the targetSdkVersion and MaxSdkVersion in AndroidManifest.xml. But it was compiled by the Android 10 in the project.properties file, which is based on Android Gingerbread (Google, 2018). The number of packages in latest edition is about two times more than the existing edition.

The other reasons for modifying the code:

1. The existing code has a number of useful functions such as debug logger and traffic logger, and some of the files of utility class could be useful for approaching development such as DateUtil.class.

2. Re-writing usually takes longer time than we expected, and there is no need to code some repetitive functions.

3. the existing UI design makes application become a mature product, and program structure is clear and understandable.

4. To be honest, I did not have any working experience in Android.

However, some drawbacks provide information about reasons for rewriting existing code. Firstly, the most important problem is that application crashing issue has not been solved. In fact, the application has a crash problem and I am trying to solve it based on application crash report. Furthermore, the kernel of the algorithm has to rewrite, ActivityMonitor.java. Even worse, it has not been provided with the effective interface. Alternatively, our workloads would be bigger than we thought. In addition, the number of the newest package in Android Oreo is twice as many as the target version.

# Conclusion

In my point of view, the refactoring code is a more suitable choice in this situation due to the mentioned reason. Refactoring is the preferred way to incrementally improve the application. It is slow paced to improve the quality of the application. Rewrite has its advantages, but it is a riskier option due to lack of working experience in Android.

In addition, the application can keep the original algorithm and add another method to measure steps, it will make application suitable for most of the edition. However, a number of other functional changes can also increase the workload.

In conclusion, the crashing problem should be resolvable easily, which may be caused by deployment. Because it has been used to collect data for research. Based on readable code and reasonable program structure, the existing source code need be refactored for updating the version and changing the kernel of arithmetic.

# References

Google, 2018. *Android API.* [Online]   
Available at: https://developer.android.com/guide/topics/manifest/uses-sdk-element.html  
[Accessed 9 2 2018].

Harries Tim, E. P. R. R., 2016. Effectiveness of a smartphone app in increasing physical activity amongst male adults: a randomised controlled trial. *BMC Public Health,* Volume 1, p. 925.

Martin Mladenov, M. M., 2009 . *A Step Counter Service for Java-Enabled Devices Using a Built-In Accelerometer.* Dublin, Proceedings of the 1st International Workshop on Context-Aware Middleware and Services: affiliated with the 4th International Conference on Communication System Software and Middleware (COMSWARE 2009).

Nagpa, V., 2016. *Android Sensor Programming By Example.* 1st ed. Birmingham: packt publishing.

# Appendix

Some important functions will be listed below:

ActivityMonitor.java: it monitors users’ activities through accelerometer sensor(useless)

DateUtil.java: Misc static date utilities

bActiveApp.java: the main entry of program and init the ACRA

AlarmReceiver.java: Reveives alarm signal that wakes the phone up

ActivityRecord.java: hold activity data for a given day

DebugLog.java: writing to debug log on the SD card

GlobalExceptionHandler.java: catch exception globally and log them to the SD card.

Globals.java: definition global variables

IActivitymonitor.java: used by ActiviityMonitor’s binder

Scheduler.java: creates threads to do routine data posting to server

ServiceManager.java: ensure service is set

TrafficCounter.java: a traffic logger for measuring bActive’s data usage over mobile networks

Updater.java: checks the web API and. APK version and downloads installation if necessary

TutorialToaster.java: check whether user has been shown toasts for the Activity

UsageRecord.java: a single record of the user entering and exiting the application

Wakeful.java: functionality to keep the phone awake and allow it to sleep

WebServiceProxy.java: sends activity records to the server

in Org.swanseacharm.bactive.database: database command and update data to serve

bActiverData.java: database command

DatabaseConstants.java

DatabaseProxy.java

UsageProxy.java

In package (org.swanseacharm.bactive.ui): allow users to view past weeks’ ,todays’ , yesterdays’ progress. Those files are for UI

In package (org.swanseacharm.receivers): allow users to receive group’s data from service.