Lab Research Documentation and Direction

Title	Lab Research Documentation and Direction v 1.0
Document Number	
Author (Organization)	Rischan Mafrur, Advanced Network Lab, CNU
Creation Date	2015-02-20
Last Modified	2015-02-25
Version	1.0
Status	First Version
Website Lab	http://netsys.jnu.ac.kr
Path of Research Data	http://netsys.jnu.ac.kr/data

Revision History:

Modified By	Date	Version	Comments
Rischan	2015-02-25	1.0	First Version

Table of Contents

General Overview Document: Level 0

Research Description

Nowadays, smartphone capability has increased significantly. Smartphone has equipped with high processor, bigger memory, bigger storage and etc. With this equipment, smartphone has capability to running complex application. Many sensor also has embedded to the smartphone. With this sensor and log capability of smartphone, we can develop many useful system or application in different domain such as healthcare (elderly monitoring system, human fall detection), transportation (monitoring road and traffic condition), personal and social behavior, environmental monitoring (pollution, weather), and etc. To develop such system, we have to collect the user personal data and then analyze it. There are two ways to collect personal data from the users based on user involvement, they are:

1. Participatory sensing
2. Opportunistic sensing

Participatory sensing means the application still need user's intervention to complete their task. The examples for such application need user to taking text input for each time period, taking picture and etc. On the other hand, opportunistic sensing means application does not need user's intervention to complete their task, users not involved in making decisions instead smart phone itself make decisions according to the sensed and stored data.

Our lab works in this field which is exploiting smartphone sensors for many of purposed. Previous research in our lab doing research which is following participatory sensing to collect user data. They have to define first and giving label to the data. Now, our lab focus on the other side which is opportunistic sensing method. We divide our problem to two branch, there are:

1. **Personal data collector application.** To get user personal data, we have to develop application which are efficient, easy to used, not bothering users, care about user privacy, battery friendly, and etc. We want to collect good quality of personal user data as much as possible, but more sensors means increase the storage, processing and energy consumption in mobile phone. So we have to research more deeply to collect the most valuable user's personal data with minimum storage, minimum processing and energy consumption in the mobile phone.
2. **Processing, visualizing, and analyzing the data.** After collecting user personal data we have to process, visualize and analyze those data. In this part, there are five main problems to solve in this area. The problems are user's privacy, data management and processing (big data issues), data analysis algorithm, data visualization and more applications based on personal data.

History of Our Research

Our lab has many of members that doing research related with exploiting smartphone sensors. In this document, we explain and describe about several previous research from our lab members who already graduated. In this document, we bring research from Vo Viet, Ali Fahmi, and Thang Hoang.

Vo Viet

Mobile-based Activity Recognition System Using Sensory Data

I. Overview:

1. Data acquisition using built-in sensors (accelerometer) of mobile devices;
2. Data preprocessing (time interpolation, noise filtering, segmentation on Y-axis);
3. Feature extraction in both time domain and frequency domain;
4. Personalized Activity recognition: Combine clustering algorithm and Support Vector Machine (SVM) classifier

II. Problems and solutions:

1. Balance accuracy and power consumption for feature extraction

Paper title Balancing Precision and Battery Drain in Activity Recognition on Mobile Phone

Appeared in 18th IEEE International Conference on Parallel and Distributed Systems (ICPDS), 2012

Dataset
- ***SCUTT-NAA***
  - 31/44 subjects with activities fully provided
  - *Sensor:* ADXL 330 accelerometer, sampling rate = 100Hz
- ***Self-constructed data***
  - *Mobile device:* Google Android HTC Nexus One
  - *Sensor:* Bosch Sensortec's 3-axis BMA 150 accelerometer
  - *Sampling rate:* 30Hz

5 volunteers, 5 categories of activity (bicycling, downstair, jogging, upstair, walking)

Data Preprocessing
- *Linear interpolation:* 100Hz (SCUTT-NAA) and 32Hz (self-constructed data)
- *Noise elimination:* Daubechies orthogonal wavelet (Db6) decomposition at level 2

Data analysis
- ***Classifier approach (SVM):***
  - *Segmentation:* 256-sample length (8 seconds) per window, overlapping 50%
- ***Matching approach (DTW):***
  - *Segmentation:* peak detection on Y-axis, 8 gait cycles per window, overlapping at 4th peak

Feature extraction
- ***Time domain feature (TF):***
  - Time gap peaks: average gap values between two consecutive peaks

  - Mean and Variance Acceleration
  - Accelerometer Energy: amount of change on a physical activity
  - Hjorth Mobility (signal mean frequency) and Complexity (deviation of the signal from the sine shape)
- ***Frequency domain feature (FFT):***
  - The first 40 FFT coefficients

Classification

- ***Classifier approach: SVM***
- ***Matching approach: DTW***

Results

- ***SCUTT-NAA:***
  - FFT feature yields better prediction accuracy than TF
  - SVM performs better than DTW
- ***Self-constructed data:***
  - TF yields better prediction accuracy and more effective computational complexity
  - SVM performs better than DTW

2. Balance accuracy and power consumption for feature extraction and classification: select appropriate sampling rate and feature set for deploying on mobile phones

Paper title Adaptive Energy-Saving Strategy for Activity Recognition on Mobile Phone

Appeared in IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), 2012

Dataset

- ***SCUTT-NAA***
  - 31/44 subjects with activities fully provided
  - *Sensor:* ADXL 330 accelerometer, sampling rate = 100Hz
- ***Self-constructed data***
  - *Mobile device:* Samsung Galaxy Note
  - *Sensor:* 3-axis K3DH accelerometer
  - *Sampling rate:* {50Hz, 17Hz, 5Hz}

4 volunteers, 5 categories of activity (bicycling, downstair, jogging, upstair, walking)

Data Preprocessing

- *Linear interpolation:* to acquire fixed interval length signals
- *Noise elimination:* Daubechies orthogonal wavelet (Db6) decomposition at level 3 (SCUTT-NAA) and 2 (self-constructed data)

Data analysis

- Gait cycle partition using peak detection on the Y-dimensional signal
- Segment length:
  - *SCUTT-NAA:* 512-sample length, no overlapping
  - *Self-constructed data:* 256-sample length, overlapping of 128 data points

Feature

- ***Time domain feature (TF):***
  - Time gap peaks: average gap values between two consecutive

extraction
  peaks
  - Mean and Variance Acceleration
  - Accelerometer Energy: amount of change on a physical activity
  - Hjorth Mobility (signal mean frequency) and Complexity (deviation of the signal from the sine shape)
- ***Frequency domain feature (FFT):***
  - The first 40 FFT coefficients

Classification
- SVM classifier with RBF kernel

Adaptive strategy
  - *Walking:* 17Hz, TF
  - *Bicycling:* 17Hz, TF
  - *Down Stair:* 17Hz, TF
  - *Jogging:* 5Hz, TF
  - *Up Stair:* 5Hz, FFT

Results
- High sampling rates normally give better prediction
- FFT coefficients perform more effective classification than TF
- Adaptive method saves a value of 28% of energy consumption compared with non-adaptive method (50Hz, TF+FFT)

3. Personalization in mobile activity recognition system: individual model needs huge training data => improve cross-people prediction scheme

Paper title Personalization in Mobile Activity Recognition System Using K-Medoids Clustering Algorithm

Appeared in International Journal of Distributed Sensor Networks, 2013

Dataset
- ***SCUTT-NAA***
  - 44 subjects, totally 1278 samples
  - *Sensor:* ADXL 330 accelerometer, sampling rate = 100Hz
- ***Self-constructed data***
  - *Mobile device:* Google Android HTC Nexus One
  - *Sensor:* Bosch Sensortec's 3-axis BMA 150 accelerometer
  - *Sampling rate:* 30Hz
  - 6 volunteers, 5 categories of activity (bicycling, downstair, jogging, upstair, walking)

Data Preprocessing
- ***SCUTT-NAA****:* to acquire fixed interval length (100 Hz and 32Hz) signal
  - *Linear interpolation:* 100Hz
  - *Noise elimination:* Daubechies orthogonal wavelet (Db6) decomposition at level 3
- ***Self-constructed data****:*
  - *Linear interpolation:* 32Hz
  - *Noise elimination:* Db6 at level 2

Data analysis
- Gait cycle partition using peak detection on the Y-dimensional signal
- Segment length:

- *SCUTT-NAA:* 512-sample length, no overlapping
- *Self-constructed data:* 256-sample length, overlapping of 128 data points

Feature extraction

- ***Time domain feature (TF):***
  - Time gap peaks: average gap values between two consecutive peaks
  - Mean and Variance Acceleration
  - Accelerometer Energy: amount of change on a physical activity
  - Hjorth Mobility (signal mean frequency) and Complexity (deviation of the signal from the sine shape)
- ***Frequency domain feature (FFT):***
  - The first 40 FFT coefficients

Activity recognition

Combine clustering algorithm with SVM classifier

1) Generate model M_A for person A;
2) Classify unlabelled samples of person B by using model M_A;
3) Cluster the labelled samples of person B by iteratively relocating the centroids by using the Euclidean distance;
4) Extract from each cluster a number of $N = K/k$ confident samples where K is given and k is the number of classes;
5) Update model M_A by using these K confident samples.

Results

- *Mobile AR system:*
  - Computational complexity on time domain is more effective than frequency domain ($O(TF) = n$, $O(FFT) = n \log n$ where n is the signal length)
- *Personalization in predefined activities:*
  - **K-means:** $K = 25$ yields optimal result in cross-people prediction, 8% accuracy increased
  - **K-medoids:** better than K-means in small sample groups (because it is more robust than K-means in the presence of noise and outliers); accuracy decreases when the number of test samples increases except for the value $K = 20$
- *Update new activities:* K-medoids performs better than K-means

Ali Fahmi

Multimodal Biometrics for Usable Authentication System Using a Smartphone

I. Overview:

1. Singlemodal biometric for user authentication system
   - Arm's flex when responding call
   - Ear biometrics
2. Multimodal biometrics for user authentication system: arm's flex and ear shape

II. Problems and solutions:

1. User authentication using arm's flex biometric

Paper title Arm's Flex when Responding Call for Implicit User Authentication in Smartphone

Appeared in International Journal of Security and Its Applications 6(3), 2012

Data acquisition

- *Mobile phone:* Pantech Sky Vega Racer
- *Sensor:* accelerometer
- 6 volunteers, 20 patterns of two categories ((1) phone picked from desk, and (2) phone picked from pocket) for each person

Classification

- *Template Matching method:* measuring similarity and thresholding
- *Similarity* = (Euclidean distance score) / (Cosine similarity score)

Result

- ***Category 1:***
  - *Classification accuracy = 87.8%*
  - *False Match Rate (FMR) = 14%*
  - *False Non-Match Rate (FNMR) = 3.3%*
- ***Category 2:***
  - *Classification accuracy = 90%*
  - *False Match Rate (FMR) = 11.3%*
  - *False Non-Match Rate (FNMR) = 3.3%*

2. User authentication using ear biometric

Paper title Implicit Authentication based on Ear Shape Biometrics using Smartphone Camera during a Call

Appeared in International Conference on Systems, Man, and Cybernetics, IEEE, 2012

Data acquisition

- *Mobile phone:* Samsung Galaxy S2
- 20 subjects, totally 80 images of size 1600x1200, cropped to 100x165 grayscale images

Data preprocessing

- Split each image into 4 quadrantal parts

Feature extraction

- Combining histogram resulted from Local Binary Pattern (LBP) and Geometric Analysis
- 61 features are obtained

Classification

- kNN classifier

Result

- Classification rate = 92.5%

3. Multimodal biometrics for authentication: arm's flex and ear

Paper title A Study on Multibiometrics derived from Calling Activity Context using Smartphone for Implicit User Authentication System

Appeared in International Journal of Contents 9(2), 2013

Idea for combination First use arm's flex, then use ear image when the phone is put near the ear in picking a call activity

Data acquisition

- *Mobile phones:* Samsung Galaxy S3, LG Optimus II, Pantech Sky Vega Racer
- *Data source:* accelerometer, gyroscope, front camera

Data preprocessing

- **Arm flex:**
  - Linear interpolation, noise filtering (2n+1-moving average filter)
- **Ear image:**
  - Divide ear image into four subregions

Feature Extraction

- **Arm flex:**
  - Segmentation: fixed length of 250
- **Ear image:**
  - Divide ear image into four subregions
  - Combining histogram resulted from Local Binary Pattern (LBP) and Geometric Analysis

Classification

- **Arm flex:**
  - *Template Matching* by using Dynamic Time Warping (DTW) distance measure (score in [0;1])
- **Ear image:**
  - kNN classifier with Euclidean distance from histogram (score in [0;1])
  - Summation of two distance score (in [0;2]) and thresholding with values $\delta \in \{1.4, 1.6, 1.8\}$

Result

- *Accuracy:*
  - $\delta = 1.4$: 95%
  - $\delta = 1.6$: 92.5%
  - $\delta = 1.8$: 87.5%

4. Thesis: Multimodal biometrics for authentication: arm's flex and ear

Title Multimodal Biometrics for Usuable Authentication System Using a Smartphone

Idea for combination First use arm's flex, then use ear image when the phone is put near the ear in picking a call activity

Data acquisition

- *Mobile phones:* Samsung Galaxy S3, LG Optimus II, Pantech Sky Vega Racer
- *Accelerometer, gyroscope:* 30 persons, 300 data in total

- *Front camera:* 30 persons, 300 images in total

Data preprocessing

- **Arm flex:**
  - Linear interpolation, noise filtering (2n+1-moving average filter)
- **Ear image:**
  - Divide ear image into four subregions

Feature extraction

- **Arm flex:**
  - Segmentation: fixed length of 250
- **Ear image:**
  - Divide ear image into four subregions
  - Combining histogram resulted from Local Binary Pattern (LBP) and Geometric Analysis, and Monogenic Local Binary Pattern (M-LBP)

Classification

- **Arm flex:**
  - *Template Matching* by using Dynamic Time Warping (DTW) distance measure (score in [0;1])
- **Ear image:**
  - kNN classifier with Euclidean distance from histogram (score in [0;1])
  - Summation of two distance score (in [0;2]) and thresholding with values $\delta \in \{1.4, 1.6, 1.8\}$

Result

- *Accuracy:* 95% ($\delta = 1.4$)
- *Receiver Operating Characteristics (ROC) analysis:* calculate area under curve (AUC)
  - AUC = 0.8731 for arm flex only
  - AUC = 0.9218 for ear only
  - *AUC* = 0.9301 when combined

Thang Hoang

Gait Authentication on Mobile Phone Using Pattern Recognition and Biometric Cryptosystem

I. Overview:

1. Data acquisition using built-in sensors (accelerometer,magnetometer) of mobile devices;
2. Data preprocessing (time interpolation, noise filtering);
3. Data analysis (gait cycle detection, pattern extraction);
4. Feature extraction in both time domain and frequency domain;
5. Classification: Machine Learning method
   - Support Vector Machine (SVM) classifier

II. Problems and solutions:

1. Data acquisition, preprocessing and classification method selection (Template Matching method vs. Machine Learning method)

Paper title Gait identification using accelerometer on mobile phone

Appeared in International Conference on Control, Automation and Information Sciences (ICCAIS), IEEE, 2012

Data acquisition
- *Mobile device:* Google Android HTC Nexus One
- *Sensor:* Bosch Sensortec's 3-axis BMA 150 accelerometer
- *Sampling rate:* 27Hz
- 11 volunteers (24 year-old), 12 laps with 26 seconds each lap for each person

Data Preprocessing
- *Linear interpolation:* to acquire fixed interval length (32Hz) signal
- *Noise elimination:* Daubechies orthogonal wavelet (Db6) decomposition at level 2

Data analysis
- Gait cycle partition using peak detection on the Z-dimensional signal

Feature extraction
- ***Time domain feature**:* average gait cycles (AGCs) is a sequence of values where one value is an average distance between one gait cycle to others (calculated by using DTW)
- ***Frequency domain features:*** the first 40 FFT coefficients form a feature vector

Classification
- ***Template Matching method:*** DTW is performed to match two AGCs templates
- ***Machine Learning method:*** SVM with feature vector is first 40 FFT coefficients

Results
- *Identification accuracy:*
  - *DTW:* 79.1%
  - *SVM:* 92.7% , additional validation is needed

2. Examining the impact of different sampling rates (from different devices) on the preprocessing steps

Paper title Adaptive Cross-Device Gait Recognition Using a Mobile Accelerometer

Appeared in Journal of Information Processing Systems 9(2), 2013

Data acquisition
- *Mobile devices:* Google Android HTC Nexus One and LG Optimus G
- *Sensor:* accelerometer
- *Sampling rates:* 27Hz (Google HTC) and 100HZ (LG Optimus)
- 14 volunteers (23~28 year-old), 12 laps with 36 seconds each lap for each person

Data preprocessing
- *Linear interpolation:* to acquire signals with fixed interval length at 32Hz and 100Hz
- *Noise elimination:* Db6 decomposition at level n ($n = 2$ for 32Hz signal and $n = 3$ for 100Hz signal)

Data analysis

- *Data segmentation* by using autocorrelation

Feature extraction

- ***Time domain features:***
  - Average maximum acceleration
  - Average minimum acceleration
  - Average absolute difference
  - Root mean square
  - 10-bin histogram distribution
  - Standard deviation
  - Waveform length
  - Time of a gait cycle
  - Gait cycle frequency
- ***Frequency domain features:***
  - First 40 FFT coefficients
  - First 40 DCT coefficients

Classification

SVM with Radial Basis Function (RBF) kernel

Classification result

99.81% (Google HTC, Db6 at level 2) and 97.53% (LG Optimus, Db6 at level 3)

Feature validation

- *Measure:* Average Error Rate (AER) and Intra-class Correlation Coefficients (ICC)
- ***Time domain features:***
  - High ICC values (0.7~0.996) => time domain features are high reliable regardless of sampling rate
  - Low AER => not influenced by the sampling rate
- ***Frequency domain features:***
  - Fair to good values of ICC (0.666~0.804) => reliable
  - High AER => very sensitive to the sampling rate

Sampling rate examination

- *Sampling rate* = {16+4k} with k = 1,2,...,21
- *Result:* best classification result with sampling rate of 32~36Hz, noise filtering at level 2

Noise filtering

Higher levels of decomposition will eliminate noise better

- *Level 1:* 12~48Hz
- *Level 2:* 12~100Hz (accuracy rate decreases when the sampling rate increases, best classification achieved at sampling rate 32~36Hz)
- *Level 3:* best accuracy rate of 97.53% at the sampling rate of 100Hz

3. Preprocessing step: Handling mobile installation issues: disorientation and misplacement of mobile phone in side the trouser's pocket

Paper title

A Lightweight Gait Authentication on Mobile Phone Regardless of Installation Error

Appeared in Security and Privacy Protection in Information Processing Systems 405:83-101, 2013 (SEC 2013)

Data acquisition
- *Mobile phone:* Google Android HTC Nexus One (sampling rate of 27Hz)
- *Sensor:* accelerometer, magnetometer
- 38 volunteers (24~28 year-old), 18 laps with 36 seconds each lap for each person, three types of footwear (sleeper, sandal and shoe)

Data preprocessing
- *Signal transformation:* rotation by using magnetometer to detect the roll, pitch and yaw angles
- *Linear interpolation:* to acquire fixed interval length (32Hz) signal
- *Noise filtering:* DB6 wavelet decomposition at level 2

Data analysis
- *Segmentation:* based on gait cycles (2,4,8)

Feature extraction
- Feature extraction in both time domain and frequency domain
- Feature subset selection by using Sequential Forward Selection (SFS) algorithm and Sequential Floating Forward Selection (SFFS) algorithm

Classification
- SVM with RBF kernel

Result
- *Accuracy:* 94.93% (SFFS)
- False-Match-Rate (FMR): 0%
- False-Not-Match-Rate (FNMR): 3.89%
- Authentication time: <4 seconds

4. System security and privacy concerns

Paper title Secure and Privacy Enhanced Gait Authentication on Smart Phone

Appeared in The Scientific World Journal, 2014

Data acquisition
- *Mobile phone:* Google Nexus One (sampling rate of 27 ± 2Hz)
- *Sensor:* accelerometer
- 34 volunteers (24~28 year-old), 18 laps for each person with different types of footwear and clothes

Data preprocessing
- *Linear interpolation:* to acquire fixed interval length (32Hz)
- *Noise filtering:* Db6 wavelet decomposition at level 2

Data analysis
- Gait cycle based segmentation

Feature extraction
- Feature extracted in both time domain and frequency domain. Feature vector is of length 290.
- Binary feature vector extraction by using quantization.
- Extract reliable bits by integrating Gaussian distribution to each components of the feature vector. Feature vector's length is reduced.

Key binding
- Randomly generate a binary secret key
- Calculate a the value by using a cryptographic hash function
- Encoding using Bose-Chaudhuri-Hocquenghem (BCH) scheme
- Binding using exclusive-OR operator

Authentication
- Decoding using BCH algorithm to obtain the secret key
- Calculate the hash value using the equivalent cryptographic hash function
- Matching between the two hash values

Result

Key length = 50 bits:
- False Acceptance Rate (FAR) = 3.92%
- False Rejection Rate (FRR) = 11.76%

Facilities

This section explain about our lab facilities which used in our research. The list of facilities that we used divided to three components are smartphone, devices, application, and server.

Smartphone

The lists of smartphones that can be used in our research.

No.	Brand/Type	OS	Count	Available
1.	Nexus 7 Wifi Only Black	Android	2	Yes

Applications

The lists of applications that can be used in our research.

No.	Applications	Type	Authors	Platform	Available
1.					
2.	App data collector	Client (mobile)	Thang	Android	Yes
3.	Data processing & visualizing platform	Server	Rischan	R shiny, linux	Yes

Server

Data Description: Level 1

We have explained about the previous research which each topic has dataset. In this section, we try to explain about the data description and collection methods. We have two kind types of data, are: individual data means the data which are collected and used by lab members to doing their research, ITRC data means we also have project from ITRC which is to collect personal user data.

Individual Data

This section explain about the dataset which used by our lab members. We have explained in previous section about the researches were done by Vo viet, Ali Fahmi, and Thang.

Vo Voiet Dataset

Ali Fahmi Dataset

Thang Dataset

ITRC Data

ITRC project collect user personal data for two times, in this section we will explain the first and the second data collected by ITRC.

First Data Collected by ITRC

The First data collected by ITRC classified by 5 types are: Application (A), Broadcast (B), GPS/location (G), Network (N), and Sensor (S). The duration of collecting data is every 15 minutes and uploading period from mobile storage to the server is every 60 minutes. The explanation of event types, values in each types can be seen below.

Event Type	Event	Value 1	Value 2	Value 3
A	Name of application	Usage duration		
B	INCOMING_CALL_END	Phone Number	Duration (sec)	
B	OUTGOING_CALL_END	Phone Number	Duration (sec)	
B	SMS_RECEIVED	Phone Number	Text len	
B	SMS_SENT	Phone Number	Text Len	
B	RINGER_MODE_CHANGED	Mode(NORMAL/SILENT/VIBRATE)	Ring Vol	Music Vol
B	CONFIGURATION_CHANGED	Mode(LANDSCAPE/PORTRAIT)		
B	bluetooth.ACL_CONNECTED	BT Name	BT Address	
G	GPS	Provider(network/gps)	Latitude	Longitude
N	SIG	Net Type(GSM/CDMA)	Signal Strength	
S	ACCELEROMETER	value 1	value 2	value 3
S	AMBIENT_TEMPERATURE	value 1	value 2	value 3
S	GAME_ROTATION_VECTOR	value 1	value 2	value 3
S	GRAVITY	value 1	value 2	value 3
S	GYROSCOPE	value 1	value 2	value 3

S	GYROSCOPE_UNCALIBRATED	value 1	value 2	value 3
S	LIGHT	value 1	value 2	value 3
S	LINEAR_ACCELERATION	value 1	value 2	value 3
S	MAGNETIC_FIELD	value 1	value 2	value 3
S	MAGNETIC_FIELD_UNCALIBRATED	value 1	value 2	value 3
S	PRESSURE	value 1	value 2	value 3
S	PROXIMITY	value 1	value 2	value 3
S	RELATIVE_HUMIDITY	value 1	value 2	value 3
S	ROTATION_VECTOR	value 1	value 2	value 3
S	SIGNIFICANT_MOTION	value 1	value 2	value 3
W	Web History	URL	Numb. Of visit	

Second Data Collected by ITRC

Current research in our lab, we focus on opportunistic sensing which is we develop application which does not need user's intervention. To analyze the result, because of the data that we collected does not have any label so we prefer to use unsupervised learning.

In this research we develop two systems are:

1. Application data collector
2. Data extraction and visualization

Application data collector is application that we used for collecting user's personal data. This application is android application. After we have all of data from the user, we have to extract and visualize, and also analyze it. To extract, visualize, and analyze the data, we use R programing language.

Application Data Collector

To develop application data collector, we do not develop from scratch, we use Funf library. The Funf Open Sensing Framework is an Android-based extensible framework, originally developed at the MIT Media Lab, for doing phone-based mobile sensing. Funf provides a reusable set of functionalities enabling the collection and configuration for a broad range of data types. Funf is open sourced under the LGPL license. Figure 1 shows Funf framework can collect many of sensing from smartphone such location, movement, communication and usage, social proximity, and many more. In this document, we do not describe details about Funf architecture but we describe about the data that we have collected and how to extract, visualize and analyze it. More details about Funf architecture can be seen in the main site of Funf[1] and also Funf developer site[2].

[1] http://www.funf.org/

[2] https://code.google.com/p/funf-open-sensing-framework/wiki/FunfArchitecture

Table 1: **List of probes and time period of recording**

No.	Probes	Interval,duration(s)
#1	Location(GPS)	300
#2	Wi-Fi	300
#3	Bluetooth	300
#4	Battery	300
#5	Call Log	86400
#6	SMS Log	86400
#7	Application Installed	86400
#8	Hardware Info	86400
#9	Contact	86400
#10	Browser Search Log	86400
#11	Browser Bookmark	86400
#12	Light Sensor	120,0.07
#13	Proximity	120,0.07
#14	Temperature	120,0.07
#15	Magnetic Field	120,0.07
#16	Pressure	120,0.07
#17	Activity Log	120,0.07
#18	Screen Status	120,0.07
#19	Running Application	120,0.07

Data Description

Our application follows opportunistic sensing because we do not want to bothering user much. To do that we must define the time (interval and duration), when the application will request the data from the smartphone. Interval means how many times in second system will send data request to the smartphone. The example, we set interval 300 seconds means 5 minutes, so application will request and store the data for every 5 minutes. Duration is used in sensor data because without duration is useless to collect the sensors data. The example of duration, when we set interval 120 seconds and duration 0.07 s so the application will send data request to the smartphone for every 2 minutes and the system will record the data during 0.07 seconds.

Table 1.shows the interval and duration from each probes. Those interval and duration already tested and we thought those setting was optimum one but we can change those setting by change the value on the string.xml in android project. Figure 2a shows the string.xml file in the directory of android project and Figure 2b shows inside the string.xml file, we can change value of interval and duration in that file.

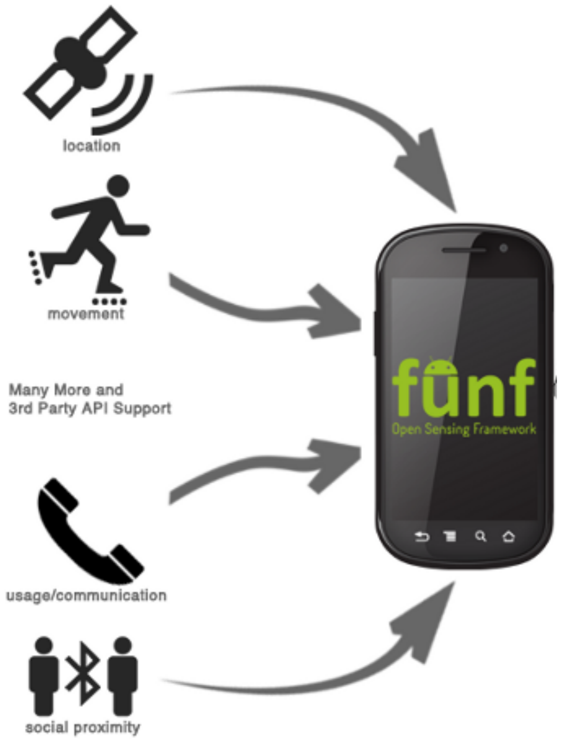
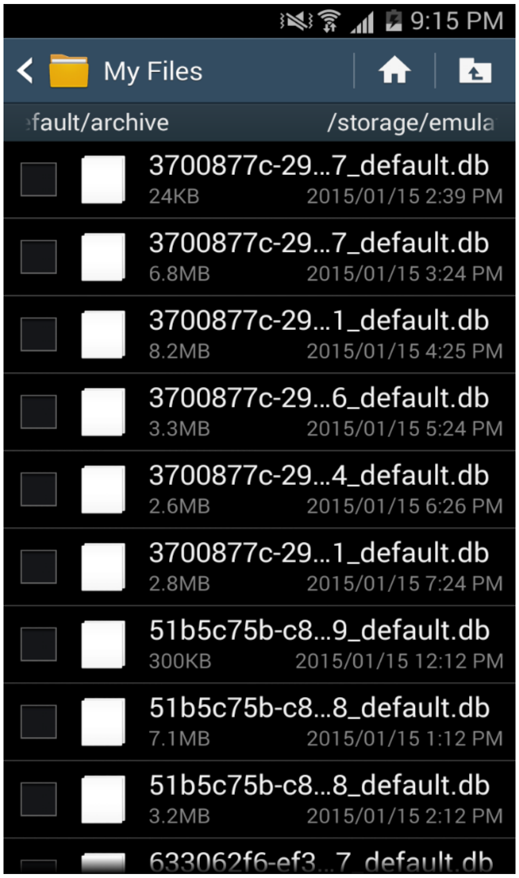


Figure 1. Funf Open Sensing Framework



fault/archive /storage/emula

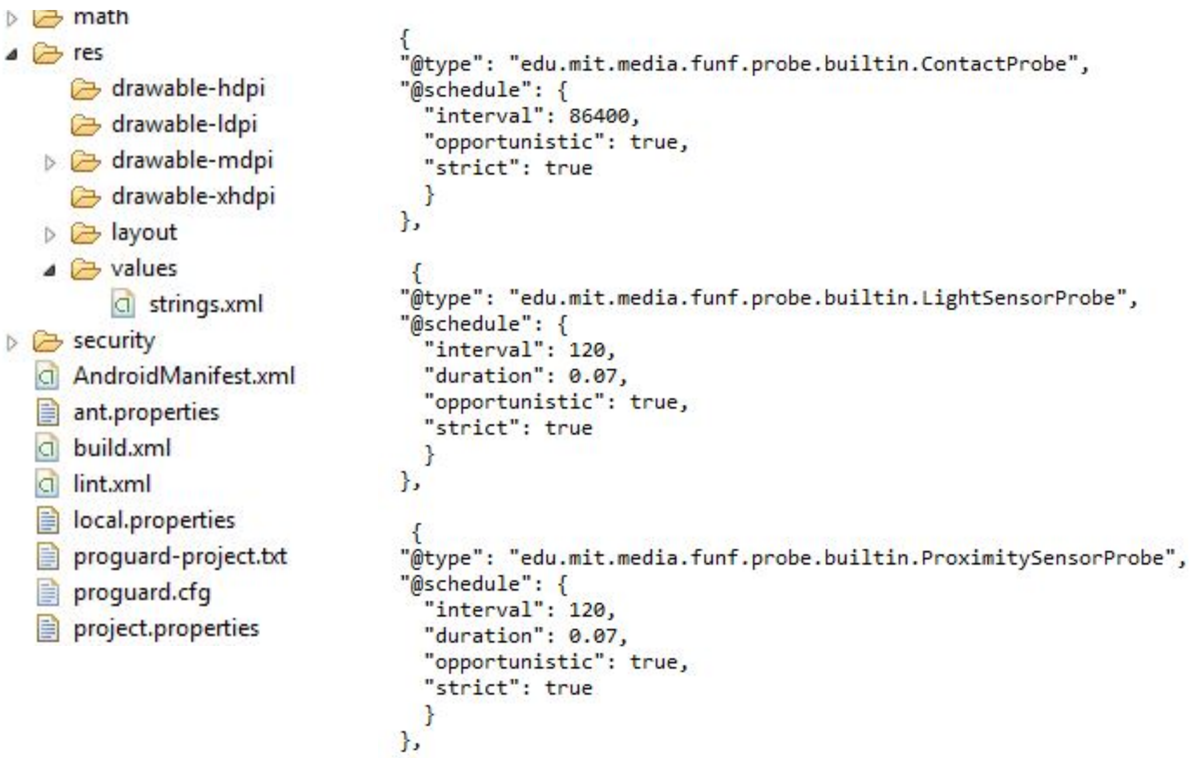
3700877c-29...7_default.db 24KB	2015/01/15 2:39 PM
3700877c-29...7_default.db 6.8MB	2015/01/15 3:24 PM
3700877c-29...1_default.db 8.2MB	2015/01/15 4:25 PM
3700877c-29...6_default.db 3.3MB	2015/01/15 5:24 PM
3700877c-29...4_default.db 2.6MB	2015/01/15 6:26 PM
3700877c-29...1_default.db 2.8MB	2015/01/15 7:24 PM
51b5c75b-c8...9_default.db 300KB	2015/01/15 12:12 PM
51b5c75b-c8...8_default.db 7.1MB	2015/01/15 1:12 PM
51b5c75b-c8...8_default.db 3.2MB	2015/01/15 2:12 PM
633062f6-ef3...7_default.db	

Figure 3. Personal data in user's smartphone

To make easy for remembering, we classify the data to three of data categorization, are:

1. On Request Data (Current Data)
2. Historical Data (Saved in Android db)
3. Continuous Data (Sensor data)

On request data means we try to ask current values from android system such as location, battery, nearby Bluetooth and etc. Historical data means the data that already store in android database so we try to access and collect it, the example of historical data are contact, call log, sms log, and etc. Continuous data means we can get those data continuously such as sensor data (accelerometer, gyroscope, magnetic field, and etc). Another important thing is because we are living in time dimension space so every data has timestamp. Funf already has features to collect time, Funf using UNIX UTC (Coordinated Universal Time) which is (Unix time or POSIX time or Unix timestamp) is the number of seconds that have elapsed since January 1, 1970. To convert UNIX time to the human readable time, we can use POSIX function in R or another programming language.



```
{
"@type": "edu.mit.media.funf.probe.builtin.ContactProbe",
"@schedule": {
  "interval": 86400,
  "opportunistic": true,
  "strict": true
  }
},

 {
"@type": "edu.mit.media.funf.probe.builtin.LightSensorProbe",
"@schedule": {
  "interval": 120,
  "duration": 0.07,
  "opportunistic": true,
  "strict": true
  }
},

 {
"@type": "edu.mit.media.funf.probe.builtin.ProximitySensorProbe",
"@schedule": {
  "interval": 120,
  "duration": 0.07,
  "opportunistic": true,
  "strict": true
  }
},
```

(a) (b)

Figure 2. (a) strings.xml file in project directory, (b) inside the string.xml file

Data that we collected using our application will be store in SQLite database format with (*.db) extension, the view of data can be seen in Figure 3. To open those database, we can use SQLite browser that can be download in SQLite browser main site[3].

Table 2. On Request Data Table

_id	name	timestamp	value (JSON)
	SimpleLocationProbe	Unix UTC	
	WifiProbe		
	BluetoothProbe		
	BatteryProbe		

On Request Data

Table 2.shows the table of On Request Data. The table contain four columns, _id is automatically generated by database engine, name means the name of probes (sensors), timestamp column is time when system store the data to the phone’s storage, and value is the value that returned from the sensors. On request data has four of probes are location, nearby Wi-Fi, nearby Bluetooth, and battery.

[3] http://sqlitebrowser.org/

Simple Location Probe

Location is one of the most important information from the user. In this research, we try to get the location information from the users. The value that returned by system is like this:

```
{"mAccuracy":1625.0,"mAltitude":0.0,"mBearing":0.0,"mElapsedRealtimeNanos":219893
72000000,"mExtras":{"networkLocationSource":"cached","networkLocationType":"cell"
,"noGPSLocation":{"mAccuracy":1625.0,"mAltitude":0.0,"mBearing":0.0,"mElapsedReal
timeNanos":21989372000000,"mHasAccuracy":true,"mHasAltitude":false,"mHasBearing":
false,"mHasSpeed":false,"mIsFromMockProvider":false,"mLatitude":35.1837595,"mLong
itude":126.9052379,"mProvider":"network","mSpeed":0.0,"mTime":1403484137091},"tra
velState":"stationary"},"mHasAccuracy":true,"mHasAltitude":false,"mHasBearing":fa
lse,"mHasSpeed":false,"mIsFromMockProvider":false,"mLatitude":35.1837595,"mLongit
ude":126.9052379,"mProvider":"network","mSpeed":0.0,"mTime":1403484137091,"timest
amp":1403484137.255}
```

That data which from location probes representing a geographic location. A location can consist of a latitude, longitude, timestamp, and other information such as bearing, altitude and velocity. All locations generated by the *LocationManager* are guaranteed to have a valid latitude, longitude, and timestamp (both UTC time and elapsed real-time since boot) and all other parameters are optional. In general, usually we only use latitude and longitude to define the human location, but in this data we have many of data, another data such as accuracy, bearing, altitude, and elapse real time are explained below.

Accuracy

Get the estimated accuracy of this location, in meters. We define accuracy as the radius of 68% confidence. In other words, if you draw a circle centered at this location's latitude and longitude, and with a radius equal to the accuracy, then there is a 68% probability that the true location is inside the circle.

Altitude

Get the altitude if available, in meters above the WGS 84 (World Geodetic System) reference ellipsoid. If this location does not have an altitude then 0.0 is returned. The coordinate origin of WGS 84 is meant to be located at the Earth's center of mass; the error is believed to be less than 2 cm.

Bearing

Get the bearing, in degrees. Bearing is the horizontal direction of travel of this device, and is not related to the device orientation. If this location does not have a bearing then 0.0 is returned.

Elapsed Real Time

Note that the UTC time on a device is not monotonic: it can jump forwards or backwards unpredictably. So always use *getElapsedRealtimeNanos()* when calculating time deltas. On the other hand, *getTime()* is useful for presenting a human readable time to the user, or for carefully comparing location fixes across reboot or across devices.

More details about the key and values from the location probes can be seen in Android API documentation through this link.

http://developer.android.com/reference/android/location/Location.html#

Nearby Wi-Fi Probe

One of important information about the user is nearby Wi-Fi. This probes will collect all of Wi-Fi information which is near with user. This data also can be collected by using our application data collector. Returned value from the system is looks like:

```
{"BSSID":"b0:c7:45:7d:0f:7c","SSID":"rischan","capabilities":"[WPA2-PSK-
CCMP+TKIP][ESS]","frequency":5180,"level":-46,"timestamp":1403476993.05}
```

We have 6 couple of keys and values, BSSID, SSID (Access Point name), capabilities, frequency, level, and timestamp.

Capabilities

Describes the authentication, key management, and encryption schemes supported by the access point.

Frequency

The frequency in MHz of the channel over which the client is communicating with the access point.

Level

The detected signal level in dBm, also known as the RSSI. Use *calculateSignalLevel(int, int)* to convert this number into an absolute signal level which can be displayed to a user.

Nearby Bluetooth Probe

Beside of nearby Wi-Fi, one of important information related with human is nearby Bluetooth. This probes will collect all of Wi-Fi information which is near with user. The value that we get from our application looks like:

```
{android.bluetooth.device.extra.DEVICE":{"mAddress":"74:F0:6D:E8:ED:67"},"android
.bluetooth.device.extra.NAME":"RRI-ITMS
PC","android.bluetooth.device.extra.RSSI":-79,"timestamp":1404128054.397}
```

We have information about the device, in this case Bluetooth device address, also Bluetooth name, RSSI, and timestamp.

android.bluetooth.device.extra.RSSI

Used as an optional short extra field in ACTION_FOUND intents. Contains the RSSI value of the remote device as reported by the Bluetooth hardware. Constant Value: "android.bluetooth.device.extra.RSSI“. More details about Bluetooth documentation can be seen in Android API documentation through this link

http://developer.android.com/reference/android/bluetooth/BluetoothDevice.html

Battery Probe

Battery Probe will collect battery information from the user’s smartphone such as charging or discharging, health condition, level, and etc. The value that returned by system looks like:

```
{"charge_type":0,"health":2,"icon-small":17303540,          "level":89,"online":1,
"scale":100,"status":3,"technology":"Li-
ion","temperature":305,"timestamp":1403476991.281,"voltage":4138}
```

We have 11 couple of keys and values from those data. The description about the meaning of values of charge_type, health, status, and voltage can be seen below:

Charge Type value meaning

- BATTERY_PLUGED_AC =1
- BATTERY_PLUGGED_USB =2
- BATTERY_PLUGGED_WIRELESS=4

Status value meaning

- BATTERY_STATUS_CHARGING =2
- BATTERY_STATUS_DISCHARGING =3
- BATTERY_STATUS_FULL =5
- BATTERY_STATUS_NOT_CHARGING =4
- BATTERY_STATUS_UNKNOWN =1

Health values meaning

- BATTERY_HEALTH_COLD =7
- BATTERY_HEALTH_DEAD =4
- BATTERY_HEALTH_GOOD =2
- BATTERY_HEALTH_OVERHEAT =3
- BATTERY_HEALTH_OVER_VOLTAGE =5
- BATTERY_HEALTH_UNKNOWN =1
- BATTERY_HEALTH_UNSPECIFIED_FAILURE =5

Voltage

Integer containing the current battery voltage level. Constant Value: "voltage".

More details about Battery documentation can be seen in Android API documentation in this link http://developer.android.com/reference/android/os/BatteryManager.html

Historical Data

Table 3.shows the table of historical data. The table contain four columns, _id is automatically generated by database engine, name means the name of probes (sensors), timestamp column is time when system store the data to the phone's storage, and value is the value that returned from the sensors. Historical data are the call log data, SMS log, the list of installed application in user's smartphone, user's smartphone device (hardware) info, bookmark in smartphone browser, Log search (history) in smartphone browser, and contact in user's smartphone.

To protect user privacy we use SHA to hash the privacy information such as user name in contact, phone number, name of caller, and etc.

Table 3. Historical Data Table

_id	name	timestamp	value (JSON)
	CallLogProbe	Unix UTC	
	SmsProbe		
	ApplicationsProbe		
	HardwareInfoProbe		
	BrowserBookmarksProbe		
	BrowserSearchesProbe		
	ContactProbe		

Call Log Probe

The data from Call log probes looks like:

```
{"_id":2172,"date":1403874310514,"duration":160,"name":"{\"ONE_WAY_HASH\":\"d5c70
34c3a03ea8ec287f7e8f082d6ec8c07ffb1\"}","number":"{\"ONE_WAY_HASH\":\"d4f6776ca77
2a1d8fadb157ef323e906d78d8d9a,"timestamp":1403874310.514,"type":2}
```

We have 7 couple of JSON keys and values, date is the date when user call (incoming/outgoing) the date in UNIX timestamp format, duration is the duration of user when he/she make call, name and number are hashed, timestamp, and type. The value of type explained below:

Type value meaning

- INCOMING_TYPE = 1
- OUTGOING_TYPE = 2
- MISSED_TYPE = 3

Sms Log Probe

The returned data from user's smartphone of the SMS log probes looks like:

```
{"address":"dad42137da1fcasdsaga54d6c0f0dc8cd1d42e7a3","body":"{\"ONE_WAY_HASH\":
\"c1f3942137da1fca36554d6c0f0dc8cd1d42e7a3\"}","body-byte-len":90,"body-token-
byte-len":"3-52-16-16-","body-token-
count":4,"date":1403316814524,"read":true,"thread_id":215,"timestamp":1403316814.
524,"type":1}
```

Similar with call log, the address, body text are hashed because this information is related to user privacy. Even though we encrypted some of information but we do not lose the pattern of information. If the address (phone number) is same the output of SHA hash also same. In this probe, we also collect the pattern of body token count, based on that data we know the length of message, the number of letters in each words, and etc.

In SMS log probes data we have key "type", the meaning of type value explained below:

Type value meaning

- MESSAGE_TYPE_ALL =0
- MESSAGE_TYPE_INBOX =1
- MESSAGE_TYPE_SENT =2
- MESSAGE_TYPE_DRAFT =3
- MESSAGE_TYPE_OUTBOX =4
- MESSAGE_TYPE_FAILED =5
- MESSAGE_TYPE_QUEUED =6

If the type message (SMS) is SENT, the data have key “status”, and the meaning of the value in key “status” explained below:

Status value meaning

- STATUS_NONE =-1
- STATUS_COMPLETE =0
- STATUS_PENDING =32
- STATUS_FAILED =64

Installed Application probe

The data from Installed application probes can be seen below, this data is only from one application data:

```
{"dataDir":"/data/data/com.lifevibes.trimapp”,"enabled":true,"enabledSetting":0,"
icon":2130837526,
"installed":true,"installedTimestamp":null,"isTrusted":0,"nativeLibraryDir":"/dat
a/data/com.lifevibes.trimapp/lib","packageName":"com.lifevibes.trimapp","processN
ame":"com.lifevibes.trimapp”,"sourceDir":"/system/app/TrimApp_phone_J.apk","targe
tSdkVersion":17,"taskAffinity":"com.lifevibes.trimapp”,"timestamp":1403476969.264
,"uid":10142}
```

Installed application probes collect the list of installed applications in user’s smartphone. The data above is an example of one data from one application. Based on that data we can determine the name of application using the name of package and that data also provides the information about the directory that used by application.

Hardware Info Probe

To know user’s smartphone specification, we can use this probes. The data from this probes looks like:

```
{"androidId":"5a0d4221916c50ce","bluetoothMac":"08:D4:2B:2A:05:3D","brand":"samsu
ng","deviceId":"354257050990298","model":"SHV-
E250K","timestamp":1403489373.257,"wifiMac":"08:D4:2B:2A:05:3E"}
```

Based on data from hardware info probes we can know the information about device Bluetooth mac, device brand, phone model, and Wi-Fi mac.

Bookmark and Log Search Probe

This application also provides the bookmark probes and log search probes. Bookmark probes will collect all of bookmark data from user's smartphone browser. Search log probes will collect all of history (search log) from user's smartphone browser. The data from bookmark probes looks like:

```
{"_id":999,"bookmark":0,"created":0,"date":1403860944022,"timestamp":1403860944.0
22,"title":"□                                                                       -
□□□□□","url":"http://portal.jnu.ac.kr/Education/Webservice_S/Default.aspx","vi
sits":125}
```

Based on that data we have information about the date in UNIX timestamp format, the title of bookmark, url, and how many user visit those bookmark.

The data from search log probes looks like:

```
{"_id":2,"date":1383925223295,"search":"facebook","timestamp":1383925223.295}
```

We have date in UNIX timestamp, the log search, and also timestamp.

Contact Probe

The data from contact probes can be seen below:

```
{"contactData":"contact_id":3,"custom_ringtone":"{\"ONE_WAY_HASH\":\"\"}","displa
y_name":"{\"ONE_WAY_HASH\":\"50bf609648d98370521094b6b724d240bd469610\"}","in_vis
ible_group":1,"last_time_contacted":0,
photo_id":0,"send_to_voicemail":0,"starred":0,"times_contacted":0,"timestamp":140
4296933.626}
```

Similar with Call and SMS log, the data which related with user privacy were hashed. From the contact data we can know the information about the name of people in contact (hashed), group, last time contacted, and many more. Even though the name of contact was hashed but we still can analyze the pattern. When the user try to contact one of person, if the name is same the output of hash also same, so we still have the pattern data, even we do not know exactly the name of people whom contacted by user.

Continuous Data

Table 4.shows the table of continuous data. The table contain four columns, _id is automatically generated by database engine, name means the name of probes (sensors), timestamp column is time when system store the data to the phone's storage, and value is the value that returned from the sensors.

Table 4. Continuous Data Table

_id	name	timestamp	value (JSON)
	LightSensorProbe	Unix UTC	{"accuracy":1,"lux":**121.0**, "timestamp":**1402725082**.124436}
	ProximitySensorProbe		{"accuracy":0,"distance":**8.0**, "timestamp":**1402725082**.030173}
	TemperatureSensorProbe		
	MagneticFieldSensorProbe		{"accuracy":2,"timestamp":**1402725082.**028829,"x":-**26**.939999,"y":**-8**.5199995,"z":**-24**.06}
	PressureSensorProbe		{"accuracy":0,"pressure":**999**.82, "timestamp":**1402725083**.016377}
	ScreenProbe		{"screenOn":**true**,"timestamp":**1402725416**.351}
	RunningApplicationsProbe		
	ActivityProbe		

Android Sensors Data

The Sensor Probes that we used are light sensor, proximity sensor, temperature sensor, magnetic field sensor, and pressure sensor. Table 4. Shows the table of continuous data and on the last column, we can see the example value from each sensor probes. The data which came from android sensor have accuracy the meaning of accuracy are described below:

Accuracy

Meaning of accuracy values in sensors data, are:

- SENSOR_STATUS_ACCURACY_HIGH means this sensor is reporting data with maximum accuracy, return value = 3.
- SENSOR_STATUS_ACCURACY_LOW means this sensor is reporting data with low accuracy, calibration with the environment is needed, return value = 1
- SENSOR_STATUS_ACCURACY_MEDIUM means this sensor is reporting data with an average level of accuracy, calibration with the environment may improve the readings, return value = 2.
- SENSOR_STATUS_UNRELIABLE means the values returned by this sensor cannot be trusted, calibration is needed or the environment doesn't allow readings, return value = 0.
- SENSOR_STATUS_NO_CONTACT means the values returned by this sensor cannot be trusted because the sensor had no contact with what it was measuring (for example, the heart rate monitor is not in contact with the user), return value = -1.

The details information about android sensors can be seen in Table 5.

Table 5. Android Sensors Explanation

Sensor	Type	Description	Common Uses
TYPE_ACCELEROMETER	Hardware	Measures the acceleration force in m/s^2 that is applied to a device on all three physical axes (x, y, and z), including the force of gravity.	Motion detection (shake, tilt, etc.).
TYPE_AMBIENT_TEMPERATURE	Hardware	Measures the ambient room temperature in degrees Celsius (°C). See note below.	Monitoring air temperatures.
TYPE_GRAVITY	Software or Hardware	Measures the force of gravity in m/s^2 that is applied to a device on all three physical axes (x, y, z).	Motion detection (shake, tilt, etc.).
TYPE_GYROSCOPE	Hardware	Measures a device's rate of rotation in rad/s around each of the three physical axes (x, y, and z).	Rotation detection (spin, turn, etc.).
TYPE_LIGHT	Hardware	Measures the ambient light level (illumination) in lx.	Controlling screen brightness.
TYPE_LINEAR_ACCELERATION	Software or Hardware	Measures the acceleration force in m/s^2 that is applied to a device on all three physical axes (x, y, and z), excluding the force of gravity.	Monitoring acceleration along a single axis.
TYPE_MAGNETIC_FIELD	Hardware	Measures the ambient geomagnetic field for all three physical axes (x, y, z) in μT.	Creating a compass.
TYPE_ORIENTATION	Software	Measures degrees of rotation that a device makes around all three physical axes (x, y, z). As of API level 3 you can obtain the inclination matrix and rotation matrix for a device by using the gravity sensor and the geomagnetic field sensor in conjunction with the getRotationMatrix() method.	Determining device position.
TYPE_PRESSURE	Hardware	Measures the ambient air pressure in hPa or mbar.	Monitoring air pressure changes.
TYPE_PROXIMITY	Hardware	Measures the proximity of an object in cm relative to the view screen of a device. This sensor is typically used to determine whether a handset is being held up to a person's ear.	Phone position during a call.
TYPE_RELATIVE_HUMIDITY	Hardware	Measures the relative ambient humidity in percent (%).	Monitoring dewpoint, absolute, and relative humidity.
TYPE_ROTATION_VECTOR	Software or Hardware	Measures the orientation of a device by providing the three elements of the device's rotation vector.	Motion detection and rotation detection.
TYPE_TEMPERATURE	Hardware	Measures the temperature of the device in degrees Celsius (°C). This sensor implementation varies across devices and this sensor was replaced with theTYPE_AMBIENT_TEMPERATURE sensor in API Level 14	Monitoring temperatures.

Running Application Probe

We have installed application probes which can collect the list of installed application in user's smartphone. To know user interest we also try to collect the current applications which user used or running application.

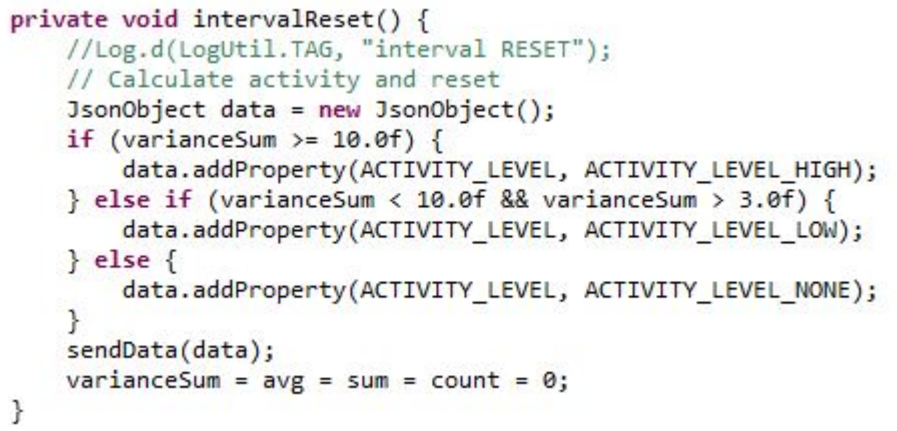
The data from running application probes looks like:

```
{"duration":6.143,"taskInfo":{"baseIntent":{"mAction":"android.intent.action.MAIN
","mCategories":["android.intent.category.LAUNCHER"],"mComponent":{"mClass":"kr.a
c.jnu.netsys.MainActivity","mPackage":"edu.mit.media.funf.wifiscanner"},
"mPackage":"edu.mit.media.funf.wifiscanner","mWindowMode":0},"id":30,"persistentI
d":30},"timestamp":1402725116.144}
```

We have information about the name of application package which is in current running also the time usage (duration).

Activity Probe

In our application we do not collet the accelerometer value directly but we use algorithm to determine the status of activities which are none, low, or high. We use sum of variance to detect the user activity based on accelerometer value. The details algorithm can be seen in Figure 4.



```
private void intervalReset() {
    //Log.d(LogUtil.TAG, "interval RESET");
    // Calculate activity and reset
    JsonObject data = new JsonObject();
    if (varianceSum >= 10.0f) {
        data.addProperty(ACTIVITY_LEVEL, ACTIVITY_LEVEL_HIGH);
    } else if (varianceSum < 10.0f && varianceSum > 3.0f) {
        data.addProperty(ACTIVITY_LEVEL, ACTIVITY_LEVEL_LOW);
    } else {
        data.addProperty(ACTIVITY_LEVEL, ACTIVITY_LEVEL_NONE);
    }
    sendData(data);
    varianceSum = avg = sum = count = 0;
}
```

Figure 4. Activity log algorithm

The value of activity probes that we get from the application looks like:

```
{"activityLevel":"none","timestamp":1402725083.715}
```

How to Describe the Data: Level 2

About Data Collection and Data Itself

About data collection and data itself in this section will explain about the name of dataset, the duration of dataset collection, how many people who participated, who was in charge for those data, and the current location of those dataset.

Individual Data

Vo Viet Dataset

No.	Name of Dataset	When (in duration)	How many people participated	Who was in charge	Data location
1.	- ***SCUTT-NAA***		31	Public dataset	

Ali Fahmi Dataset

No.	Name of Dataset	When (in duration)	How many people participated	Who was in charge	Data location

Thang Dataset

No.	Name of Dataset	When (in duration)	How many people participated	Who was in charge	Data location

ITRC Data

As we mention in previous that ITRC project has two times of collecting personal user data, in this section we explain about those kinds of data.

ITRC First Data Collection

Number of students: 38 students
Time (duration): 2013/10/21 - 2013/11/21
Location path: C:\ITRC_DATA

Sensors and Events	Number of rows
SMS_SENT	3631
SMS_RECEIVED	5575
INCOMING_CALL_END	3850
OUTGOING_CALL_END	3991
Application □□ □□	549796
Sensor log	725617
ACCELEROMETER	126416
PROXIMITY	98928
GYROSCOPE	95426
MAGNETIC_FIELD	134157
ORIENTATION	60362
PRESSURE	23851
LIGHT	23524
Web Browser	41944
GPS	103105
Bluetooth	3098

ITRC Second Data Collection

We have finished to collect user personal data for the ITRC second data collection project. The all of data stored in Rischan PC using path C:\ITRC_DATA. The data has been archived in zip format with name "itrc 2014 userdata finalpoint 20140903.zip" with size 4.25 GB. The extracted data can be accessed in path C:\ITRC_DATA\itrc 2014 userdata finalpoint 20140903\. The size of all of data after extracted is 28.7 GB. Extracted data contain 47 directories in different name for each student data. We have tied to looking information about those data such as the size of data from each student and starting point also ending point of data recording. The result of data summarization which contain with name of directories, size, starting point, and ending point can be seen in Table 6.

Table 6. Data Summarization from 47 students.

No.	Data ID	Size (MB)	Starting Point	Ending Point
1.	ENFP_0719	628	6/30/2014 8:26	8/20/2014 0:18
2.	ENFP_0773	664	6/26/2014 12:34	8/18/2014 4:58
3.	ENFP_2012	661	6/27/2014 6:11	9/2/2014 3:57
4.	ENTJ_5868	6890	6/27/2014 5:31	8/13/2014 0:00
5.	ENTJ_6454	121	6/26/2014 5:32	8/6/2014 18:53
6.	ENTJ_6966	272	7/2/2014 7:24	8/19/2014 11:22

7.	ENTP_5623	455	6/30/2014 4:49	8/19/2014 20:57
8.	ESFJ_2301	145	6/27/2014 5:31	8/20/2014 2:58
9.	ESFJ_9284	158	6/26/2014 12:34	8/18/2014 4:58
10.	ESFP_0912	278	6/26/2014 5:28	8/18/2014 8:53
11.	ESFP_3295	-		
12.	ESFP_4634	486	6/27/2014 5:25	8/20/2014 4:10
13.	ESFP_7467	607	6/26/2014 5:27	8/19/2014 7:18
14.	ESTJ_0371	2390	7/3/2014 16:21	8/16/2014 21:03
15.	ESTJ_3022	183	6/26/2014 5:28	8/18/2014 23:22
16.	ESTJ_5071	1920	7/2/2014 2:34	9/11/2014 1:49
17.	ESTJ_5190	258	7/30/2014 6:04	8/24/2014 1:43
18.	ESTJ_5824	173	6/26/2014 5:29	8/18/2014 3:51
19.	ESTJ_6510	756	6/27/2014 5:30	8/20/2014 8:09
20.	ESTP_4301	232	6/26/2014 5:29	8/20/2014 4:39
21.	ESTP_5154	990	6/27/2014 5:31	8/13/2014 0:00
22.	INFP_1993	432	6/26/2014 5:31	8/20/2014 0:31
23.	INTJ_5498	342	6/26/2014 5:28	8/20/2014 2:49
24.	INTJ_7906	312	6/14/2014 11:00	8/16/2014 23:01
25.	INTP_3739	1030	6/27/2014 5:28	8/18/2014 5:58
26.	INTP_6399	199	6/26/2014 5:29	8/12/2014 8:32
27.	INTP_9712	180	6/26/2014 5:37	8/16/2014 18:05
28.	ISFJ_2057	183	6/27/2014 5:32	8/14/2014 23:19
29.	ISFJ_2711	767	7/31/2014 0:51	8/20/2014 6:59
30.	ISFJ_7328	133	6/30/2014 7:09	8/19/2014 23:37
31.	ISFP_4030	2380	6/27/2014 6:11	9/2/2014 3:57
32.	ISFP_4282	613	6/27/2014 5:27	8/20/2014 2:46
33.	ISTJ_0178	158	6/26/2014 5:28	8/19/2014 5:05
34.	ISTJ_0386	284	6/26/2014 5:27	8/19/2014 7:18
35.	ISTJ_2068	339	6/26/2014 5:29	8/18/2014 5:30
36.	ISTJ_2837	186	6/27/2014 5:27	8/22/2014 5:41
37.	ISTJ_3052	131	6/27/2014 5:27	8/20/2014 3:41
38.	ISTJ_4659	325	7/2/2014 2:34	9/11/2014 1:49
39.	ISTJ_4667	156	6/26/2014 5:29	8/15/2014 10:44
40.	ISTJ_4700	170	7/3/2014 6:50	8/25/2014 13:08
41.	ISTJ_4753	363	6/26/2014 5:29	8/18/2014 23:48
42.	ISTJ_4968	95	7/3/2014 16:21	8/16/2014 21:03
43.	ISTJ_9139	473	7/3/2014 16:21	8/20/2014 5:57
44.	ISTJ_9576	198	7/4/2014 1:00	8/18/2014 7:12
45.	ISTP_3948	500	6/26/2014 5:29	8/20/2014 1:28
46.	ISTP_7676	365	6/27/2014 5:31	8/19/2014 22:11
47.	XXXX_XXXX	434	6/27/2014 5:31	8/21/2014 6:02