*UNIVERSITY OF MAURITIUS RESEARCH JOURNAL – Volume 21 – 2015*
*University of Mauritius, Réduit, Mauritius*

# MoBateriE: A Personalised Profile-Based Intelligent and Adaptive Energy Manager

**D Jaguessur ***
*Department of Computer Science and Engineering*
*Faculty of Engineering*
*University of Mauritius,*
*Réduit*
*E-mail: djroshnie@gmail.com*

**S Pudaruth**
*Department of Computer Science and Engineering*
*Faculty of Engineering*
*University of Mauritius,*
*Réduit*
*E-mail: s.pudaruth@uom.ac.mu*

**J Seetohul**
*Department of Computer Science and Engineering*
*Faculty of Engineering*
*University of Mauritius,*
*Réduit*
*E-mail: jeeten@uom.ac.mu*

*Paper accepted on 11 May 2015*

## Abstract

Increasing electronic waste has forced the mobile phone industry to move into a new era of energy consumption awareness. Trends show that mobile phone manufacturers are implementing more and more features on a single device. Using these features has led to an increase in electricity consumed by mobile phones, thus forcing users to charge them more frequently. Studies have shown that one can maximise a phone's battery life by limiting the number of charging cycles (Columbus, 2013). In this study, a survey was conducted in order to collect mobile phone usage behaviour. Data concerning most frequently used mobile features together with their usage frequency were collected. Based on the information gathered, a mobile application, *MoBateriE*, was designed. This application consists of an expert system which at first studies users' behaviour and later imitates them whenever needed. *MoBateriE* has been designed for smart phone users who want to make extensive use of the features available on their phones without exhausting the battery efficiency of the phone. This application was distributed to a large number of users. Usage logs of thirty-three users were collected and analysed. Results obtained show that after using *MoBateriE*, users enjoyed a four per cent increase in usage time per charging cycle.

***Keywords:*** *electronic waste, battery life, charging cycle, mobile application*
***Running Title****: MoBateriE – Mobile Power Management System*

**For correspondences and reprints*

## 1. INTRODUCTION

Protecting the environment is everyone's concern. Necessary precautions need to be taken in order to minimise wastage and carbon emissions in the environment. Innovations, decrease in prices, and obsolescence have resulted in a growing surplus of electronic waste around the world. Technical solutions are available, but in most cases, frameworks, collections of subsystems or logistics need to be implemented before a technical solution can be applied (UNEP, 2009).

Statistics show that about fifty million tons of electronic wastes are thrown away each year. The USA discards around thirty million computers yearly whereas in Europe, one hundred million phones are disposed of each year (UNEP, 2009). Studies have shown that people are upgrading their televisions, mobile phones, audio equipment, computers and printers more frequently than ever. Mobile phones and computers are replaced most often.

Figure 1 shows the average number of months it took for citizens from different countries to replace their mobile phones over four different years. The countries have been selected for their geographic and economic diversity. In general and quite surprisingly, mobile phone users are replacing their mobile phones less often. In 2007, the mean replacement cycle was 39 months with a standard deviation of about 15 months, while in 2010, the mean replacement cycle was 45 months with a standard deviation of about 20 months. Americans replaced their mobile phones only after 22 months in the year 2010. British and Koreans were not far behind. On average, people from Brazil, Israel and Finland replaced their mobile phones after more than 6 years. Developed countries like Germany, Italy and Japan have relatively low mobile replacement rate. Thus, it is very difficult to draw a general conclusion for all the countries. The trends are highly country-specific.

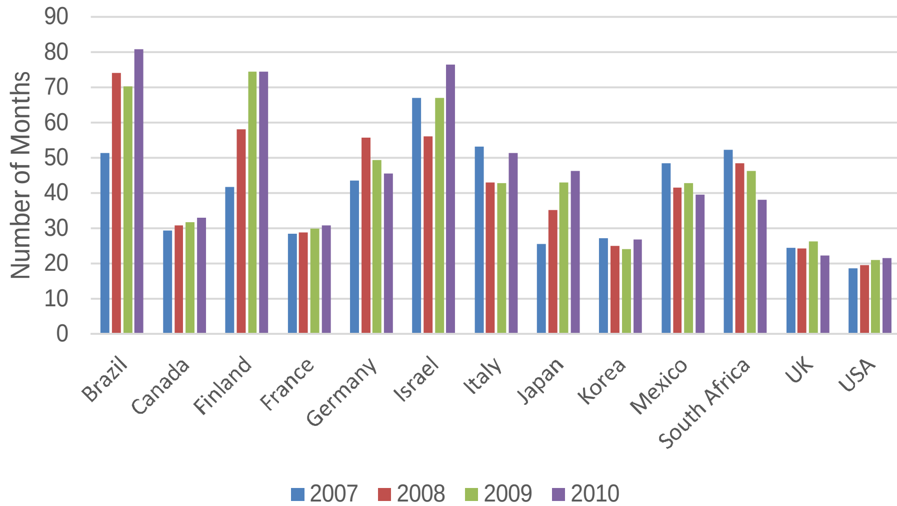


**Figure I: Mobile Phone Replacement Cycle per Country**

More recent surveys done in 2013 show that people are sticking to their phone-upgrade behaviour trends. Many users in the developed world are upgrading their mobile phones only after two years. There are still a significant proportion of users who feel the need to replace their mobile phones on an annual basis. These are normally fashion and technology driven buyers who need to always have the newest handset in their hands (Entner, 2013).

In Mauritius, the trend is similar to the United States. Mauritians are changing their mobile phones every two years (Kowlesser & Bokhoree, 2010). According to data published by Statistics Mauritius (CSO, 2012), the number of mobile phone users in Mauritius has increased by 8.7% in the year 2011, that is from 1,190,900 in 2010 to 1,294,100 in 2011. This change can be explained by factors like innovations, wear and tear and market trends. Studies have shown that people replace their mobile phones because their current devices do not have up to date functionalities like a high resolution camera and wireless connections. Sometimes due to unfortunate circumstances, like accidents, mobile devices are no longer usable and need to be replaced. Moreover, mobile devices are nowadays made in such a way that they have a limited number of charging cycles. Once this number is attained the device becomes unusable.

## 2. LITERATURE REVIEW

Ferdous and Poet (2012) investigated on the impact of using various Identity Management Systems on power consumption of a mobile phone, especially when accessing this service via different browsers. They use different Identity Management Systems to access the same set of services and investigate its impact on the power consumption. In order to quantify their research, they make use of the PowerTutor application. The latter uses built-in battery voltage sensors and the knowledge of battery discharge behaviour to record real-time power consumption estimates of different components. However, this study only shows that when different browsers are used to access the same data online, the amount of energy consumed differs. No solution has been proposed yet.

Liebergeld *et al.* (2012) used virtualisation technology to increase machine utilisation and power. They designed a power management infrastructure that can enforce power management on mobile devices. They also developed a model which takes into consideration both the hardware characteristics and the constraints of usage scenarios. However, for their tests, they have used mobile devices which have hardware CPU virtualization capabilities. No tests have been done on other devices.

Sivaramakrishnan and Kailarajan (2011) implemented a circuit that can receive transmitted radio frequency signals from other devices and extract power from them. The radio frequency signals are received by an antenna and streamed through a rectifier circuit and then through a power converter circuit, which increases the voltage. The power produced can be used to recharge batteries. This is a revolutionary idea, however, no consideration has been taken about the impact on the environment. These radio frequency signals transmitted will need to be transferred via a terminal, which will shelter huge transmitters. These transmitters will consume far more energy and the impact on environment will be greater.

Barbeau *et al.* (2011) proposed a location aware framework, which supports real-time applications for GPS-enabled mobile phones. It dynamically adjusts platform parameters for performance and conserves battery life. This framework has been designed only for mobile phones, which are GPS-enabled. It cannot be used on tablets or laptops.

Aaron and Heiser (2010) analysed the power consumption of the Openmoko Neo Freerunner mobile phone. They developed a power model of the device in order to analyze the energy consumption and battery lifetime under a number of usage trends. They concluded that the most effective power management approach on mobile devices is to shut down unused components and disable their power supplies whenever possible. All the tests were done on the Openmoko Neo Freerunner mobile phone. Generalising the results might not be appropriate as not all mobile phones use the same technologies and components. Moreover, the usage patterns which have been cited in the paper have been decided by the authors' themselves based on their own experiences. No elicitation methods have been used to get this information. For *MoBateriE*, a survey was done to analyse usage history and various tests have been done on several versions of Android. Furthermore, *MoBateriE* can be used on tablets as well as phones unlike the power model designed by Aaron and Heiser.

Wang *et al.* (2009) implemented the *Energy Efficient Mobile Sensing System (EEMSS)* on Nokia N95 devices that uses sensor management scheme to manage built-in sensors on the N95. This system includes a GPS, a Wi-Fi detector, an accelerometer and a microphone in order to recognise daily human activities. They proposed and implemented several algorithms for accelerometer and microphone readings that work in real-time and this led to good performance. To evaluate their system, they selected users from two universities, and the results show that they were able to provide a high level of accuracy for a state recognition gain of around seventy-five percent of the device lifetime. EEMSS has been tested only on the N95, that is, on the Symbian operating system, which is in the fourth position in the market share after Android, IOS and Windows Phone (Mobile Statistics, 2012).

Banerjee *et al.* (2007) conducted a study on battery usage and recharge behaviour. Based on these findings an energy management system, named

Llama, had been developed. They claimed that the system design would exploit the battery energy in a user-adaptive and user-friendly way to better serve the user. However, only ten laptops and ten mobile phones were used in this study.

Irani *et al.* (2007) examined two mechanisms for battery energy saving, one when the phone is in an idle state and the other when it is in the active state. For the active state, the speed of executing a task should have been defined. However, this study did not take into consideration the transition time from one state to another and the possible power consumption during this transition.

Nishkam *et al.* (2006) designed a new context-aware battery management architecture for mobile devices known as CABMAN. This architecture consists of a battery lifetime predictor which makes use of a metric that is independent of battery age and takes into account the applications' battery usage. In this paper, the authors have considered telephony as a crucial application and expect that users always want to be able to use this application. However, other application like navigation may also be deemed to be crucial in particular circumstances. It is the users who must choose which applications they want to prioritise.

*Turducken* is a mobile device architecture which enables always-on availability and extended battery lifetime. *Turducken* integrates different mobile computing platforms, operating at different power levels, into a single device that can operate at any power level. While the system supports all of the functionalities of its highest power subsystem, it can utilize lower power subsystems to execute simpler tasks (Sorber *et al*., 2005). Such a technology requires specialised hardware which may be costly for the users to buy. Portability of such a system can be very complex. In contrast, *MoBateriE* is a free application and runs perfectly on several versions of Android.

**Related Works**

There exist several applications on the market which can be used to monitor and enhance the battery usage time of a mobile device. Table 1 below depicts a thorough comparison of some of these applications.

**Table I: Comparison of Mobile Applications**

| Application/ Functionalities | Llama Location Profiles | Tasker (Premium Version) | Battery Drain Analyzer Monitor | AVG Battery Saver & Time Up | Juice Defender | Du Battery Saver & Switch Widget | Easy Battery Saver | One Touch Battery Saver | Advanced Mobile Care | Smart Optimization Box | Battery Doctor | Battery Stats Plus | Smart Settings | MoBateriE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Free | x | | | x | | x | x | x | | | x | x | x | x |
| Displays percentage | | x | x | x | | x | x | x | x | x | x | x | | x |
| Display Remaining Time | | x | x | x | | x | x | | x | x | x | x | | x |
| Time based Profile Switcher | x | x | x | | x | x | x | | | x | x | | x | x |
| Context based Profile Switcher | x | x | x | x | x | x | x | x | x | x | x | | x | x |
| Stage Charging | | | | | | x | | x | | | x | | | |
| Power Saving Profile | x | x | x | | x | x | | x | | x | x | | x | x |
| User-defined Profile | x | | | x | x | x | x | x | x | x | x | | x | x |
| Pre-defined Profiles | x | x | x | | x | x | x | x | | x | x | | x | x |
| Toggles between 2G/ 3G/ 4G | x | | | | x | x | | | | x | | | | x |
| Takes Decision | | | | | | | | | | | | | | x |

*Llama Location Profiles* allows a user to control his phone's features during particular parts of the day, depending on where he is at any given time. It uses the mobile phone network masts to determine his position whereas *MoBateriE* uses GPS location. Using phone network masts are not as precise as GPS location. A single network cell covers a large area, so the location defined can be inaccurate. Another disadvantage of this application is the fact that network cells are not a perfect way of getting a location. As networks change, there is a chance that the list of locations for the cells become obsolete.

*Tasker* is an application for Android which performs sets of actions based on the application’s time, date, location and event. *Du Battery Saver & Switch Widget* shows battery level, battery temperature and battery remaining hours on users’ phone. Compared to *MoBateriE*, these applications’ features and user interfaces are more complex. Newcomers can easily get lost in all the functionalities that they possess. User reviews show that it takes several hours to actually configure and make them run. Also they do not guarantee battery savings.

*Battery Drain Analyzer Monitor* shows battery status and battery life statistic for the last thirty days. The analyzer also displays battery power consumed by individual applications. *Easy Battery Saver* saves battery by intelligently dealing with the phone's network connectivity, screen time-out and screen brightness. *One Touch Battery Saver* has a battery monitor which shows both status of battery life and usage. When the power is insufficient in the device, users can click on the power saving mode to activate it. Compared to *MoBateriE*, these applications have fewer features. The number of profiles are also limited and there are no user defined profiles. Users can only set a timer to activate a profile, but no starting and ending times can be defined for the profiles.

*AVG Battery Saver & Tune Up* sets an alert to let the user know when their batteries levels are going down so users can disable specific functions in order to save power. It only displays information to users, and no actions are taken. *Battery Stats Plus* identifies and reports battery draining applications. These applications only display relevant information to the users. No actions are taken to save battery usage or increase its performance. However with *MoBateriE*, if the user is not present, that is if the phone is locked, the application takes decisions itself.

*Advanced Mobile Care* is an Android security and performance optimization application. It performs virus scans at regular intervals. It also has a game speeder, a battery saver, a call blocker, an application manager, a task killer, a privacy locker, a privacy advisor and cloud backup of contacts and logs. This application has all ideal concepts but with limited customisation. Compared to *MoBateriE*, this application is complex to use. It has so many features that the user can easily get lost in the configurations. It has a game speeder which consumes lots of energy and memory.

*Smart Optimization Box* prolongs standby time by starting in eco-mode together with brightness control. Its system management helps to speed-up smart phones. However, it does not take screen size into consideration. It is designed to fit only on wide screens. When viewed on a small screen, the screen data are partly cut off. Moreover, it has only one eco-mode which can be used to activate/deactivate settings.

*Battery Doctor indicates* how long the battery will last under different situations like playing games and using Wi-Fi. *Smart Settings* is a tool which is used to automate and optimise device settings for power savings. *Juice Defender* manages components, like 3G connectivity and Wi-Fi. Compared to *MoBateriE*, these applications do not have an expert system and thus if the phone is locked, no action is taken. They only display information on the screen while waiting for the user to take actions.

**System Design**

An energy saving Android application has been developed arising from this research. Selecting Android as operating system has been done based on recent studies made by the Worldwide Smartphone Operating System Market Share (2013). The latter demonstrate that during the first quarter of 2013, the Android Operating System and the Apple iOS (previously iPhone Operating System) were the two most used mobile operating systems. Together they represent more than ninety percent of market share in the first quarter of 2013. Android Operating System alone represents about seventy-five per cent. As iOS seems to grow very slowly, the Android market share, which is growing very quickly, seems to be the best target environment to reach the largest number of users.

The following tools have been used to develop this application: Eclipse Juno together with the Android Developer Tool plug-in, Android Software Development Kit Tools, Android Platform-tools, Android platform, Android system image for the emulator and the Java Development Kit 6.

**Front End Specifications**

The Main window displays in real time a summary of the actual battery state and status of other battery consuming system applications like Wi-Fi, GPS and phone network. Within this window, the user can access the global application preferences (standard Android preferences menu) and three main options configuration panels via clickable buttons. Global options window displays a set of global user preferences for the application, such as language, sending statistics and state check frequency.

**Code behind Specifications**

The system behind is separated in two main parts. There is a main application which handles all the interfaces and a service which is launched at device startup and handles the expert system. Furthermore, this service logs necessary information about the device usage which is then used by the expert system for decision making.

Android system has native broadcasting functionalities. These are used to manage the state of the different battery consuming services on the device like Wi-Fi, GPS and mobile network. The expert system checks the states of the different services in order to retrieve the selected configurations for manual, timeline and location profiles. It then follows a logical algorithm in order to decide the best configuration to use. All the required information such as the defined profiles, timelines and locations are stored in an SQLite3 application database. This database is automatically created at application start-up, based on an XML file which describes its structure.

**Main Application**

Main application is a standard Android application made up of several activities which is displayed on the screen. A java class is used to handle all automated tasks needed for the application to work.

**Main Activity**

The main activity is the entry point of the application. It is the first window that is displayed to the user on the screen. As a Java class, it extends the standard Activity Android class (*android.app.activity*) and implements the Android public static interface *OnClickListener* (*android.view.View.OnClickListener*) to handle the user clicks on the screen (buttons and other views). It also checks if the AI service is active otherwise it is set on.

**Broadcast Receivers**

Broadcast receivers are instantiated by the different services of the program to retrieve the usage statistics of the device in real-time. It is a multiple instance receiver, so several parts of the program could call it at the same time. A parameter is passed by the caller, to tell the broadcast receiver which device service it must monitor.

**Implementation of the Mobile Application**

**Profiles Module**

The Profiles Module allows the user to choose between different profiles that may be switched on or off manually (one at a time) when desired. There are also predefined profiles, which users will be able to customise. It must be noted that these profiles are derived based on the responses obtained from the survey. Additionally users have the possibility to create their own profiles.

**Profiles Manager**

The Profiles Manager displays a set of preset profiles such as Home, Office and Outdoor that are editable. It displays a button allowing the user to access a new window in which a custom profile can be created or modify an existing one.

**Timelines Module**

The Timelines Module gives the possibility to the user to automatically activate a profile based on the day of week and hour of the day.

**Location Module**

The Location Module is based on the GPS. The user can set several known GPS locations and then decide which profile to use when the device has reached one of the registered locations. The time interval of the GPS checks if the location has changed in order to update the distance between the two locations.

**AI (Artificial Intelligence) Module**

The AI service is the heart of the application. It is a background service running independently from the main thread. This service is used to call the broadcast receiver when the application is closed and it monitors the state of the different battery consuming services of the device. This service is able to store its actual parameters once the phone switches to power saving mode and retrieves the configuration when the device is back to its normal state. With the different profiles and parameters set by the user, the expert system establishes the configuration which is more battery-friendly for a given time and location. For example, if the user has defined that the Wi-Fi should be on at a given time and if it finds that no known or open Wi-Fi networks are available, it will stop trying to retrieve a network after a predefined number of tries. It also decides when to switch the device to power saving mode and when to activate certain settings. All automated decisions have a lower priority than human settings. The AI starts by checking the battery level; if it is lower than 10% and the device is locked then it will determine the profile to be set, else it will prompt the user to activate the critical profile. The AI service uses a decision tree based on user configurations and settings. When no setting matches the actual situation/location of the device, the AI service decides on its own which settings to activate.

**Critical Profile**

This profile is used mainly when the battery level has reached below 10%. When this profile is activated, a regular check is done to see if the charger is plugged. As soon as the charger is plugged on and the battery started charging, this profile is deactivated and the optimal profile is used. It must be noted that in this profile Wi-Fi, GPS and Bluetooth connections are disabled.

In case the battery level reached 5% and the charger is not plugged on, the airplane mode is activated and all alarms are deactivated. The algorithm given which follows explains in detail how the AI service works.

```
START
IF (Battery level <= 10) THEN
    IF phone is locked THEN
        IF AUTOCONFIG on THEN
            IF user is at a defined location THEN
                IF current time is defined THEN
                    Activate time-based profile defined
                ELSE
                    Activate location-based profile defined
                ENDIF
            ELSE
                Activate optimal profile
            ENDIF
        ELSE
            Activate critical profile
        ENDIF
    ELSE
        Display message to ask user to activate Critical Profile
        IF User press ok THEN
            Critical Profile is activated
        ENDIF
ELSE
    IF AUTOCONFIG on THEN
        IF user is at a defined location THEN
            IF current time is defined THEN
                Activate time-based profile defined
            ELSE
                Activate location-based profile defined
            ENDIF
        ELSE
            Activate optimal profile
        ENDIF
    ENDIF
ENDIF
END
```

**Optimal Profile**

When this profile is activated, the expert system checks thoroughly all the established connections (Wifi, GPS, and Bluetooth). If anyone is found connected but not in use, it is immediately disconnected. Moreover, if a connection is available but the system is unable to connect to it, the latter will stop trying to connect after three failed attempts. The optimal profile does not only cut off services which are not in use but also detects the noise level around the device and use this information to make adjustments to the profile parameters. The algorithm given below explains in detail how this profile works.

```
Start
IF optimal profile activated THEN
        IF WIFI on THEN
                WHILE (connectionCounter  < 4)
                        Search Wi-Fi connection
                        IF connection found THEN
                                Connect to WIFI
                                Break
                        ENDIF
                        connectionCounter ++
                ENDWHILE
                Set WIFI off
        ELSE
                Set WIFI off
        ENDIF
        IF GPS on THEN
                WHILE (connectionCounter  < 4)
                        Search GPS connection
                        IF connection found THEN
                                Connect to GPS
                                Break
                        ENDIF
                        connectionCounter ++
```

```
        ENDWHILE
        Set GPS off
    ELSE
        Set GPS off
    ENDIF
    IF BLUETOOTH on THEN
        WHILE (connectionCounter  < 4)
            Search devices to be paired
            IF (connection found AND device already in paired list) THEN
                Connect to device
                Break
            ELSE
                Send pairing request
                IF request accept
                    Connect to device
                    Break
                ENDIF
            ENDIF
            connectionCounter ++
        ENDWHILE
        Set BLUETOOTH off
    ELSE
        Set BLUETOOTH off
    ENDIF

Record environment noise level using microphone
final double TALK_NOISE_LEVEL = 60.0;
final double STREET_NOISE_LEVEL = 70.0;
final double CAR_NOISE_LEVEL = 80.0;
final double MACHINE_NOISE_LEVEL = 85.0;
final double CROWD_NOISE_LEVEL = 95.0;
final double MAX_NOISE_LEVEL = 100.0;

    IF(dNoiseLevel <= TALK_NOISE_LEVEL) THEN

        iRingerAverageVolume = (int) Math.round(iRingerMaxVolume
                                *(dNoiseLevel/100));
        iNotificationAverageVolume = (int)Math.round
                                (iNotificationMaxVolume* (dNoiseLevel/100));
        iMediaAverageVolume = (int) Math.round(iMediaMaxVolume *
                                (dNoiseLevel/100));
```

```
        iVoiceCallAverageVolume = (int) Math.round
                                (iVoiceCallMaxVolume *  (dNoiseLevel/100));
        iSystemAverageVolume = (int) Math.round(iSystemMaxVolume
                                 * (dNoiseLevel/100));
        iAlarmAverageVolume = (int) Math.round(iAlarmMaxVolume *
                                (dNoiseLevel/100));

ENDIF

IF (dNoiseLevel > TALK_NOISE_LEVEL AND dNoiseLevel < =
STREET_NOISE_LEVEL) THEN
        iRingerAverageVolume = (int) Math.round(iRingerMaxVolume * (65/100));
        iNotificationAverageVolume = (int) Math.round(iNotificationMaxVolume *
                                        (65/100));
        iMediaAverageVolume = (int) Math.round(iMediaMaxVolume * (65/100));
        iVoiceCallAverageVolume = (int) Math.round(iVoiceCallMaxVolume *
                                      (65/100));
        iSystemAverageVolume = (int) Math.round(iSystemMaxVolume * (65/100));
        iAlarmAverageVolume = (int) Math.round(iAlarmMaxVolume * (65/100));
ENDIF

IF(dNoiseLevel > STREET_NOISE_LEVEL && dNoiseLevel <=
CAR_NOISE_LEVEL) THEN
        iRingerAverageVolume = (int) Math.round(iRingerMaxVolume * (75/100));
        iNotificationAverageVolume = (int) Math.round(iNotificationMaxVolume *
                                        (75/100));
        iMediaAverageVolume = (int) Math.round(iMediaMaxVolume * (75/100));
        iVoiceCallAverageVolume = (int) Math.round(iVoiceCallMaxVolume *
                                      (75/100));
        iSystemAverageVolume = (int) Math.round(iSystemMaxVolume * (75/100));
        iAlarmAverageVolume = (int) Math.round(iAlarmMaxVolume * (75/100));
ENDIF

IF (dNoiseLevel > CAR_NOISE_LEVEL && dNoiseLevel <=
MACHINE_NOISE_LEVEL) THEN
        iRingerAverageVolume = (int) Math.round(iRingerMaxVolume * (80/100));
        iNotificationAverageVolume = (int) Math.round(iNotificationMaxVolume *
                                (80/100));
        iMediaAverageVolume = (int) Math.round(iMediaMaxVolume * (80/100));
        iVoiceCallAverageVolume = (int) Math.round(iVoiceCallMaxVolume *
                                (80/100));
        iSystemAverageVolume = (int) Math.round(iSystemMaxVolume * (80/100));
```

```
            iAlarmAverageVolume = (int) Math.round(iAlarmMaxVolume * (80/100));
    ENDIF
    IF(dNoiseLevel > MACHINE_NOISE_LEVEL && dNoiseLevel <=
    CROWD_NOISE_LEVEL) THEN
            iRingerAverageVolume = (int) Math.round(iRingerMaxVolume * (85/100));
            iNotificationAverageVolume = (int) Math.round(iNotificationMaxVolume *
                                    (85/100));
            iMediaAverageVolume = (int) Math.round(iMediaMaxVolume * (85/100));
            iVoiceCallAverageVolume = (int) Math.round(iVoiceCallMaxVolume *
                                    (85/100));
            iSystemAverageVolume = (int) Math.round(iSystemMaxVolume * (85/100));
            iAlarmAverageVolume = (int) Math.round(iAlarmMaxVolume * (85/100));
    ENDIF

    IF(dNoiseLevel > CROWD_NOISE_LEVEL && dNoiseLevel <=
    MAX_NOISE_LEVEL) THEN
            iRingerAverageVolume = (int) Math.round(iRingerMaxVolume * (95/100));
            iNotificationAverageVolume = (int) Math.round(iNotificationMaxVolume *
                                    (95/100));
            iMediaAverageVolume = (int) Math.round(iMediaMaxVolume * (95/100));
            iVoiceCallAverageVolume = (int) Math.round(iVoiceCallMaxVolume *
                                    (95/100));
            iSystemAverageVolume = (int) Math.round(iSystemMaxVolume * (95/100));
            iAlarmAverageVolume = (int) Math.round(iAlarmMaxVolume * (95/100));

    ENDIF

    IF(dNoiseLevel > MAX_NOISE_LEVEL) THEN
            iRingerAverageVolume = iRingerMaxVolume;
            iNotificationAverageVolume = iNotificationMaxVolume;
            iMediaAverageVolume = iMediaMaxVolume;
            iVoiceCallAverageVolume = iVoiceCallMaxVolume;
            iSystemAverageVolume = iSystemMaxVolume;
            iAlarmAverageVolume = iAlarmMaxVolume;
    ENDIF

END
```

## 3. RESULTS AND DISCUSSIONS

The application developed for this project, *MoBateriE*, was distributed freely to more than one hundred persons, out of which fifty-two people download the application from the link that was provided via email and installed it in their mobile devices. Users were informed beforehand that all their activities would be monitored and stored in logs on their own mobile devices. Subsequently, these logs will be transferred to a server once the users were connected to the Internet through a Wi-Fi connection. Only incremental logs were sent to the server. However, the logs are not removed. This is because the expert system uses logs stored in this particular file in order to take decisions. Users were also briefed about the main functionalities of the application like the possibility to create custom profiles. They were also told that they should as far as possible continue to use their mobile device in the same way as they had been doing so far and that they should not deactivate the application for the next forty days. Out of the fifty-two users whose logs we collected, only a sample of thirty-three users was taken due to the fact that a common period of at least 21 days was needed in order to give sufficient time for the AI to learn the user's behaviour.

### Charging hours efficiency

Charging hours of the first seven days were compared with those of the last seven days. The mean charging hours of the initial days was 1.49 hours whereas that of the last seven days was 1.43 hours.  Thus, giving us an overall energy efficiency of 4.03% which is highly significant.

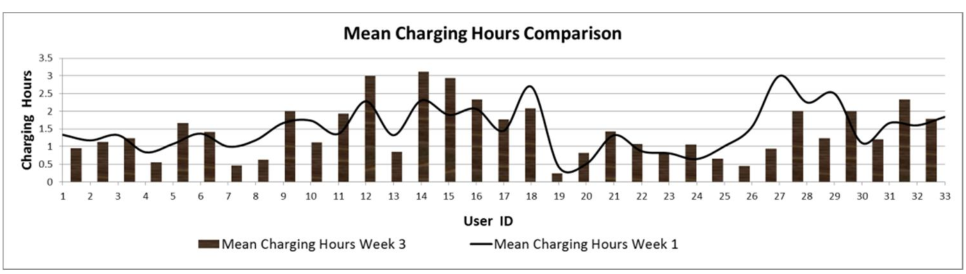


**Figure IV - Charging Hours Comparison**

From Figure IV, we note that there has been energy savings from 15 (45.5%) users. This shows that the AI has contributed in energy savings to a large number of users. The energy usage for the first week and the third week remained the same for 9 (27.3%) users. It is possible that some users were already aware of the power consumption of certain features and thus they switched them off after use. For the remaining 9 (27.3%) users, there was an increase in the energy requirements. Users whose energy usage increased after using *MoBateriE,* can be explained by the fact that they had never used a GPS service before.

### Charging cycles efficiency

Usage logs comparison also revealed that for the first seven days, users were charging their mobile devices an average of 3.42 times per week. However, we note that for the last seven days, users had charged their mobile devices only 3.33 times on average. This leads to an overall gain of around 2.65%.

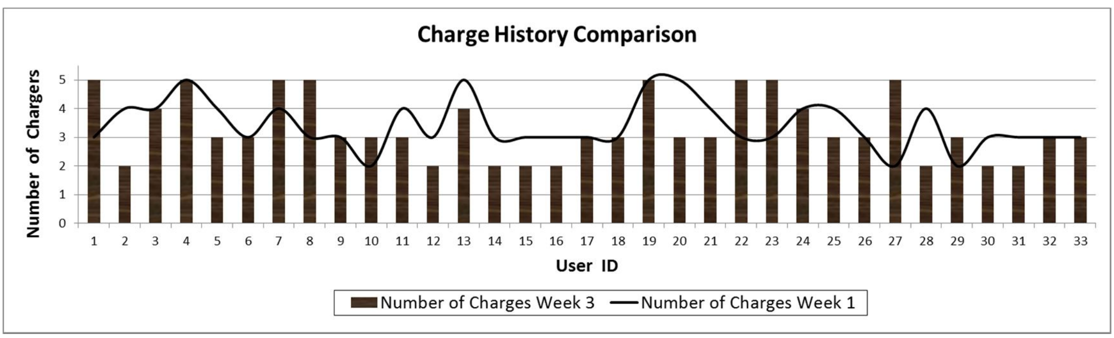


**Figure V - Charging Cycles' Comparison**

Figure V shows that the number of charging cycles has fallen to 14 (42.4%) between the first week and the third week. This again shows that the AI has been quite effective. The number of charging cycles has remained the same for 11 (33.3%) users. This could be the same group of energy-aware users who were already using their mobile devices in an energy-efficient manner. And finally, there is an increase in the number of charging cycles for 8 (24.2%) users.

## 4. CONCLUSION

E-waste is fast becoming an alarming problem in Mauritius as well as other countries around the world. Manufacturers of mobile devices are making batteries which has a limited number of charging cycles after which the mobile devices becomes almost unusable and the user has to buy a new one. A literature review was conducted about the different techniques available to minimize energy usage on mobile devices. A detailed comparison of available applications on the market had also been done. An Android-based mobile application was then developed with a view to increase the battery duration of a particular charging cycle by minimising waste of battery energy through the judicious management of services running on the device. This is in an attempt to increase the battery life, hoping that this will result in a decrease in e-waste. By using this application, we showed that users on average benefitted from a four percent (4%) gain per charging cycle. The number times a phone was charged per week has also decreased slightly. One of the major functionalities of *MoBateriE* depends on the GPS availability. However, in certain location, there are obstructions and no direct lines of sight with the satellites can be obtained (Fante and Vaccaro, 1998). In such cases, this functionality will not work properly. *MoBateriE* stores usage logs on the device internal memory for the expert system to study. Over time, this log will become bigger and bigger, thus requiring more space. One solution is to keep only recent logs as these will be more representative of the user’s behaviour. In order to effectively manage power consumption in a mobile phone, a power simulator must be built. This power simulator should map all the components of a mobile phone, study how they interact among themselves and quantifies their power consumption. Some beneficial features like staged charging and voltage information can be implemented in *MoBateriE* so as to make it even more efficient. Some green guidelines or tips can also be displayed to the users.

## 5. REFERENCES

AARON, C., & HEISER, G., 2010. *An Analysis of Power Consumption in a Smartphone*. In Proceedings of the 2010 USENIX Conference, pp. 10-21, California, USA.

BANERJEE, N., RAHMATI, A., CORNER, M., ROLLINS, S., & ZHONG, L., 2007. *Users and Batteries: Interactions and Adaptive Energy Management in Mobile Systems*. Lecture Notes in Computer Science, Vol. 4717, pp. 217-234, 2007.

BARBEAU, J. Sean, GEORGGI, N. Labib, LABRADOR, A. Miguel, PEREZ, A. Rafael, PEREZ, J. Alfredo, & WINTERS, L. Philip, 2011. *A Location-Aware Framework for Intelligent Real-Time Mobile Applications.* University of South Florida, United States.

BRAY, T., 2012. *Say Goodbye to the Menu Button*. [online] Available from: http://android-developers.blogspot.com/2012/01/say-goodbye-to-menu-button.html [Accessed 26 August 2014].

CENTRAL STATISTICS OFFICE (CSO), 2012. *Economic and Social Indicators: Information and Communication Technologies (ICT) Statistics – 2012* [online] Available from: http://www.gov.mu/portal/site/cso/menuitem.046106bcbc449145c18d5c10a0208a0c/?content_id=9c3892edfa898310VgnVCM1000000a04a8c0RCRD [Accessed 22 June 2014].

CHEDID, W., & YU, C., 2003, *Survey on Power Management Techniques for Energy Efficient Computer Systems*. Technical Report, Department of Electrical and Computer Engineering Cleveland State University, Cleveland, United States.

COLUMBUS, J., 2013. *What Is the Life Span of a Cell Phone Battery?* [online] Available from: http://www.ehow.com/about_6678887_life-span-cell-phone-battery_.html#ixzz2csYR5ZTE. [Accessed 02 August 2014].

ELMER, M., 2013. *How to use Tasker* [online] Available from: http://www.androidauthority.com/how-to-use-tasker-199872/ [Accessed 02 August 2014].

ENTNER, R., 2013. *Handset replacement cycles haven't changed in two years, but why?* [Online] Available from: http://www.fiercewireless.com/story/entner-handset-replacement-cycles-havent-changed-two-years-why/2013-03-18#ixzz2dBVVD4mm [Accessed 27 July 2014].

ENVIRONMENTAL PROTECTION AGENCY, 2011. *Electronics Waste Management in the United States through 2009*. EPA, EPA 530-R-11-002.

FANTE, R. & VACCARO, J., 1998, *Cancellation of Jammers and Jammer Multipath in a GPS Receiver*, IEEE AES Systems Magazine, Vol. 13, pp. 25-28, November 1998.

FERDOUS, S. & POET, R., 2012. *Analysing power consumption of different browsers & identity management systems in mobile phones.* International Journal of Distributed and Parallel Systems (IJDPS), Vol. 3, No. 2, March 2012.

FISHER, J., 2013. *Cell Phone Energy Consumption: The Numbers May Surprise You* [Online] Available from: http://www.renewableenergyworld.com/rea/blog/post/2013/03/cell-phone-energy-consumption-the-numbers-may-surprise-you [Accessed 18 July 2014].

FLINN, J., 2001. *Extending Mobile Computer Battery Life through Energy-Aware Adaptation.* Thesis (PhD). Carnegie Mellon University, United States.

GECKO, L., 2013, Without GPS, how does Google and Android know where I am. Available from: http://blog.geckolandmarks.com/how-does-google-and-android-know-where-i-am-w-35219 [Accessed 02 August 2014].

GREENPEACE, 2013, *Point of No Return* [online]. Available from: http://www.greenpeace.org/international/Global/international/publications/climate/2013/PointOfNoReturn.pdf [Accessed 11 July 2014].

GREENPEACE, 2005, *The E-Waste Problem* [online]. Available from: http://www.greenpeace.org/international/campaigns/toxics/electronics/the-e-waste-problem [Accessed 10 January 2014].

HAMEED, S., WOLF, A., ZHU, K. & FU, X. 2012. *Evaluation of Human Altruism Using a DTN-based Mobile Social Network Application.* In Proceedings of GI-Jahrestagung, pp. 81-95, 2012.

INTERNATIONAL DATA CORPORATION, 2013. *Worldwide Quarterly Mobile Phone Tracker*. INTERNATIONAL DATA CORPORATION, 16 May 2013.

IRANI, S., SUKLA, S. & GUPTA, R., 2007, *Algorithms for Power Savings.* ACM Transactions on Algorithms, Vol. 3, Issue 4, November 2007.

JADHWANI, D., AGRAWAL, M. & MANDE, H., 2012, *Study of Efficient Utilization of Power using green Computing.* IJACR, Vol. 2, Issue 4, December 2012.

KANTER, R., 2012. *Ten Reasons People Resist Change*. [online] Available from: http://blogs.hbr.org/kanter/2012/09/ten-reasons-people-resist-chang.html [Accessed 25 August 2014].

KOWLESSER, P. & BOKHOREE, C., 2010. *E-Waste Quantification & Characterisation for Mauritius.* [online] Available from: http://www.gov.mu/portal/sites/GreenIT/downloads/Ewaste.pdf [Accessed 01 October 2014].

LIEBERGELD, S., LANGE, M., & NEDOSPASOV, D., 2012, *Power Management on Mobile Devices with Virtual Machines.* Technical University of Berlin and German Telekom Innovation Laboratories. Germany.

MOBILE STATISTICS, 2012. Current Market Share – June 2012 [online]. Available from: http://www.mobilestatistics.com/mobile-statistics [Accessed 13 August 2014].

NISHKAM, R., JAMES, S., & LIVIU, I., 2006, *Context-aware Battery Management for Mobile Phones: A Feasibility Study*. Technical Report, Department of Computer Science, Rutgers University, USA.

RECON ANALYTICS (RA), 2011. *International Comparisons: The Handset Replacement Cycle.* United States: Recon Analytics.

ROBINSON, B., 2009. *E-waste: An assessment of global production and environmental impacts.* Science of the Total Environment (Elsevier), Volume 408, Issue 2, pp. 183–191, December 2009.

SANKAT, R., 2013. *E-warriors: In search of e-waste.* [online] Available from: http://www.e-warriors.byethost31.com [Accessed 02 August 2014].

SIVARAMAKRISHNAN, A., & KAILARAJAN, J., 2011, *A Highly Efficient Power Management System for Charging Mobile Phones using RF Energy Harvesting*. International Journal of Information Technology Convergence and Services (IJITCS), Vol 1, No, 5, October 2011.

SORBER, J., BANERJEE, N., CORNER, M., & ROLLINS, S., 2005, *Turducken: Hierarchical Power Management for Mobile Devices*. In Proceedings of the 3rd International Conference on Mobile Systems, Applications and Services (MobiSys '05), pp. 261-274, New York, USA.

SURVE, A., KHOMANE, A. & CHEKE, S., 2013, *Energy Awareness in HPC: A Survey*. International Journal of Computer Science and Mobile Computing (IJCSMC), Vol. 2, Issue 3, pp. 46-51, March 2013.

UNEP, 2009, *Recycling – from e-waste to resources*. [online] Available from: http://www.unep.org/PDF/PressReleases/E-Waste_publication_screen_FINALVERSION-sml.pdf [Accessed 22 June 2014].

WANG Y., LIN J., ANNAVARAM M., QUINN A., HONG J., KRISHNAMACHARI B., SADEH N., 2009. *A Framework of Energy Efficient Mobile Sensing for Automatic User State Recognition*. Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys 2009), pp. 179-192.