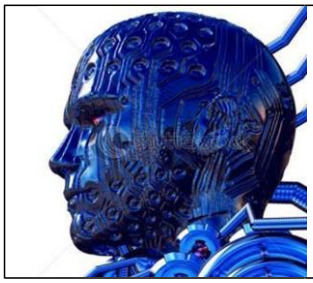

*TRS

Technology, Networks and Society

e-planning | networks | e-learning | e-government

Internal Report TRS 05/2019

Título

Reviewing the Efficiency of Current Power-Saving Approaches Used Among Different Stages of an Android-Application Lifecycle

Author(s)

Abdullah Mahmoud Almasri, Prince Sultan U.

Luís Borges Gouveia, UFP

Month, year

July, 2019

Local de presença Web http://tecnologiaredesesociedade.wordpress.com

Repositório de trabalho científico *trs http://bdigital.ufp.pt/handle/10284/3787

Universidade Fernando Pessoa

Praça 9 de Abril, 349

4249-004 Porto, Portugal

Reviewing the Efficiency of Current Power-Saving Approaches Used Among Different Stages of an Android-Application Lifecycle

Abdullah Mahmoud Almasri[1] and Luis Borges Gouveia[2]

[1] *Prince Sultan University, Riyadh, Saudi Arabia, 37582@ufp.edu.pt*

[2] *University Fernando Pessoa, Porto, Portugal, lmbg@ufp.edu.pt*

Abstract.

A common issue that is shared among android smartphones users was and still related to saving their batteries power and to avoid the need of using any recharging resources. A big number of researches were conducted in the general field of "Saving Energy in Android Smartphones". Another big number of researches were also conducted in the subfield of "Saving Energy in Android Smartphones at the Application Layer". Both fields did generate a good amount of proposed methodologies, models, frameworks and algorithms that were provided as market products or approaches. However, this review will focus only on the applications layer and the main role of this layer in saving the power of an android smartphone's battery. A review of the relevant existing literature is provided herein specifically covering various energy-saving techniques and tools proposed by various authors for Android smartphones.

Keywords: Android smartphones, Android applications, Power-saving, Android Application Lifecycle, energy efficiency

1. Background

Smartphones have grown to become constant companions to humans as they are considered to offer indispensable help in easing the daily life of individuals.

They are largely supported by numerous and diverse applications which help in for instance, directing us to our destinations, storing tickets when we travel, facilitate communications with friends and family, and entertain with videos or music. Due to the underlying importance of these mobile smart devices, there have been increasing concerns, particularly from users, regarding battery-drain which puts limitations on their usage.

Based on the existing literature, a significant share of power consumption in these smart devices is largely caused by applications that are installed on the devices (Taleb et al, 2013; Li, Tran &

Halfond, 2014). Depending on the applications' functionality, they entail activities such as data downloading, content display, and use of built-in-sensors such as GPS (Global Positioning System) related sensors. There are various components of mobile smart devices that facilitate the above activities including; GPS sensors, device' display, the CPU, and network interfaces among others.

Consequently, activities/functions of different Android smartphone applications increase the energy consumption of any of the above-mentioned components. As a result, there has been a lot of effort in the existing literature geared towards identifying and investigating the underlying potential for energy savings in relation to these smartphone applications at applications layer and OS layer levels (Moamen & Jamali, 2015; Zhang, et.al., 2010).

2. Pre-requisites of the Review

2.1 Identify the Average Android-Application Lifecycle

In order to demonstrate the main issues with current power-saving approaches, first we proposed creating a lifecycle that shows the main average stages of an average android application. The proposed cycle is shown in Figure.1:

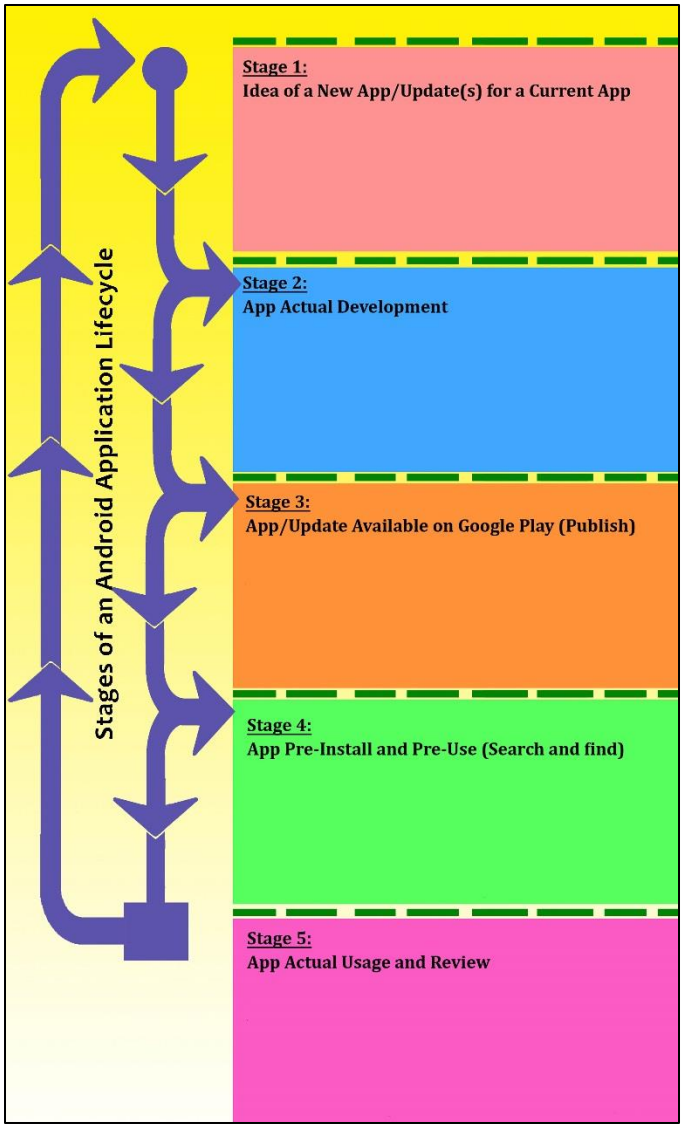

Figure .1 Average Android-Application Lifecycle

2.2 List the Concerned Parties and Identify their Involvement

Following the previous step In order to demonstrate the main issues with current power-saving approaches, we list the parties which are involved in our android application lifecycle as shown in figure.2:

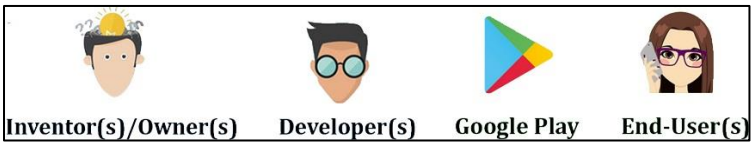

Figure .2 Parties Involved in an Android Application Lifecycle

The next item to demonstrate is the involvement of the parties among the different stages of our android application lifecycle. The involvement is described in figure.3:

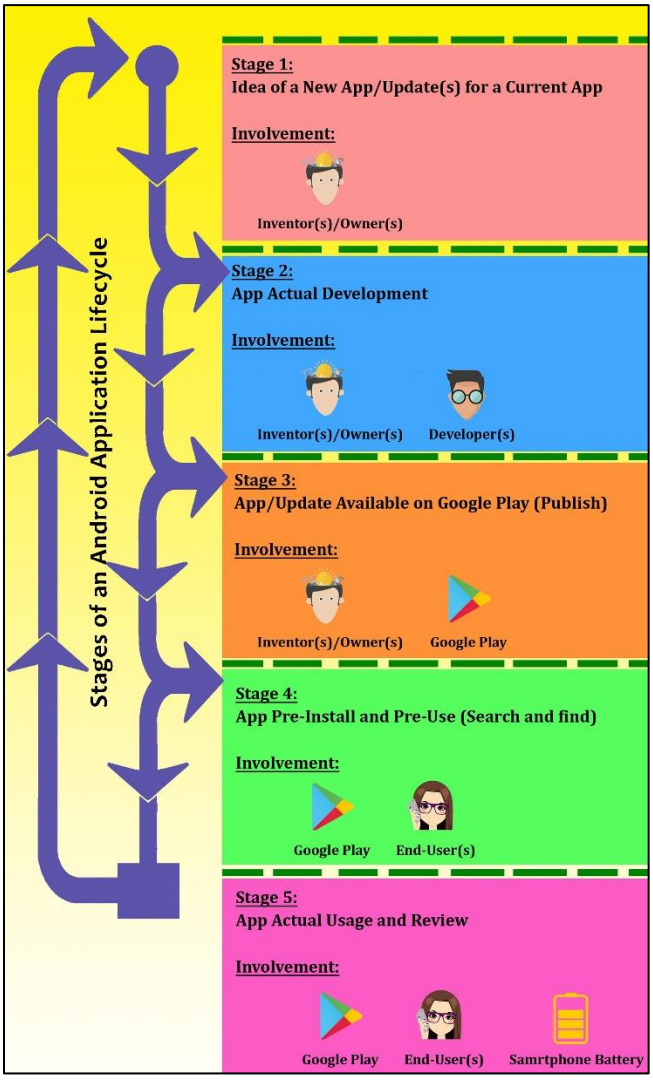

Figure .3 Involvement of Parties Among the Different Stages of an Android Application Lifecycle.

2.3 Identify the Status of an Android Application Among Different Stages of an Android Application Lifecycle

Following the above, we need to List the main statuses of an android application in terms of its presence in an android smartphone also among the different stages of our android application lifecycle, the two main statuses were Outside the End-User's Phone (Under development or Available on Google Play) or Inside the End-User's Phone (Installed & Running). Figure.4 will map the above statuses to the different stages of our android application lifecycle:

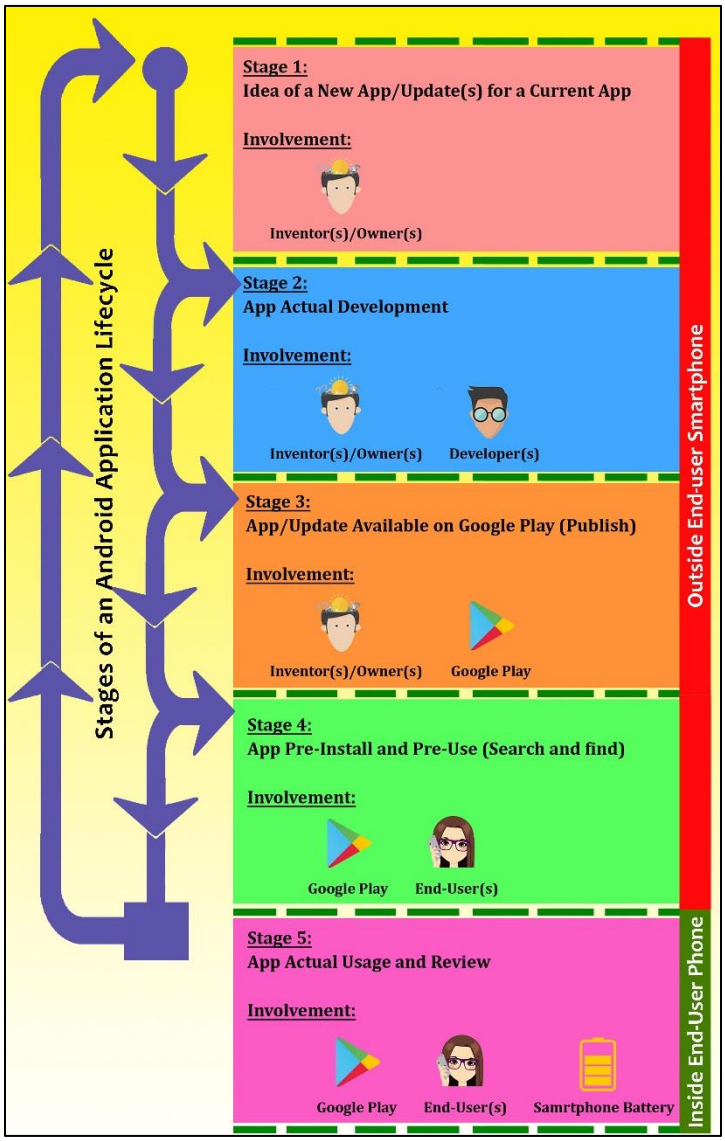

Figure .4 Status of an Android Application among Different Stages of an Android Application Lifecycle

2.4 Identify the Main Current Power-Saving Approaches

In order To summarize the current power saving approaches that are used in today's smartphones the following classification were made: Approach 1, follows the philosophy of "Simulate and estimate" the power consumption of and android application before making it available for end-user(s) by using techniques that may include but not limited to green coding, energy-aware designs, smartphone batteries simulators, historical analytical data..etc. Approach 2, follows the "Monitor, detect and control" philosophy, so it applies this on the behavior of an android application while it is running on an end-users phone and optimizing

the power consumption. Approach 3, is more about Sacrifice smartphones technology or performance by switching off a number of features for the sake of saving power philosophy.

2.5 Show the usage of current power-saving approaches among the stages of the android application lifecycle

The next stage is to show the usage of the above approaches among the stages of the android application lifecycle and from the above, Approach 1 is used in stage 2 of our android application lifecycle and involves the app inventor(s), the app developer(s) and the Android Development Platform(s), while Approaches 2 and 3 are used in stage 5 of our android application lifecycle and involve Google Play, The End-user(s) and The End-user's phone(s). The usage is shown more clearly in Figure .5

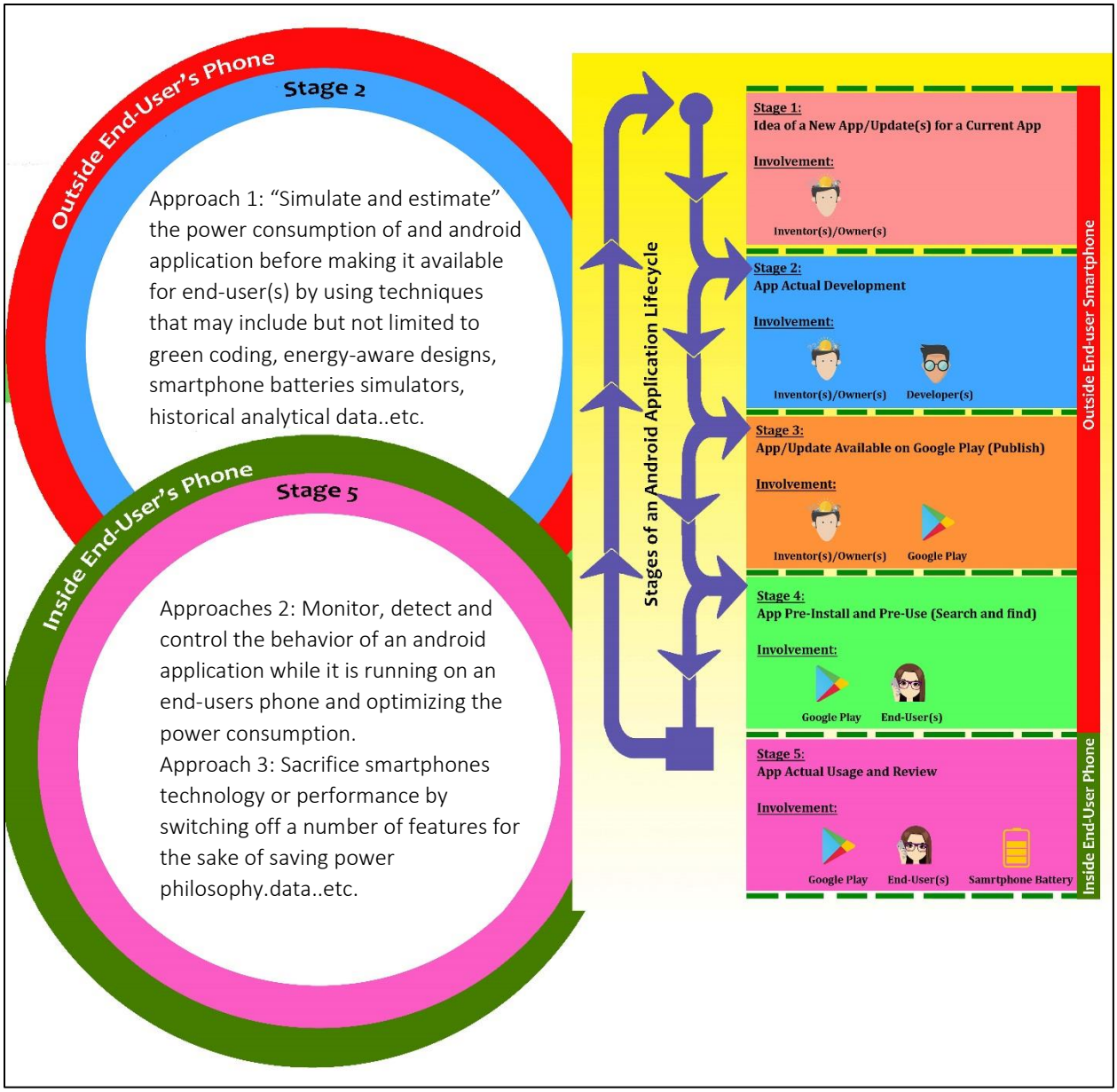

Figure .5 Usage of Current Power-Saving Approaches among the Stages of the android application lifecycle

3. The Review

3.1 Estimate and simulate power consumption approach

Westfield & Gopalan (2016) contribute towards finding a solution towards power saving techniques in smartphones through proposing an approach called Orka. According to Westfield & Gopalan (2016), the Orka approach works by providing feedback to developers of software used in smartphones. The proposed approach is designed to provide feedback on the basis of API usage

by an application as well as providing feedback on the usage of energy of the application, down to the level of the method used (Westfield & Gopalan, 2016).

The authors of the study believe that it is relatively important that energy usage of software is not disassociated from energy usage of the hardware, hence Orka is designed to generally provide feedback on the consumption of energy as a result of usage of hardware (Westfield & Gopalan, 2016). Orka carries out tests on the app through using an execution trace that is dynamically created and generated through a test script that is provided by the developer of the application. In addition, the authors suggest that the proposed Orka performs the analysis on the hardware running on emulators instead of running on physical devises (Westfield & Gopalan, 2016). Orka pulls estimations of internal energy from the emulator, after running the application, in order to provide feedback on the basis of the different components utilised.

Using the energy consumption data/metrics provided by the Orka approach, the developer of the application can make adjustments to their code in order improve the energy efficiency of their application. According to Westfield & Gopalan (2016), Orka was designed specifically for applications installed on the Android Operating System (OS). Despite the fact that Orka appears to operate in a similar manner as energy profiling solutions presented in the existing literature, Westfield & Gopalan (2016) suggest that Orka's independence from the hardware makes it different from other energy profiling systems/solutions. However, it is worth noting that, the approach used in the study does not necessarily make readings on the basis of battery discharge and it does not attempt to estimate accurately an application's energy usage.

Wang, et.al., (2017) are concerned with the energy testing stage of the app development as they believe that applications developers ought to understand both, the rate of energy consumption of their applications and the underlying reason why energy is consumed by the application. In their paper, Wang, et.al., (2017) propose E-Spector as a potential online based tool/method the inspects energy usage, visualises the application's energy consumption online in a manner that is instant, and it can also inform the developer what happened behind each hotspot of energy on an energy curve. According to Wang, et al., (2017), E-Spector mainly relies on static analysis and the instrumentation of the application to collect the underlying activities in real time from the execution of an application.

These activities are then presented on an instant energy curve in such a way that the user is able to recognise what actually took place behind each spike in energy usage (Wang, et al., 2017). The authors believe that their proposed solution is particularly more beneficial because it does not require hardware meters like many other solutions in order to calculate instant the power figures for each application at runtime since it is an online-based software solution/power model (Wang, et.al., 2017). Furthermore, Wang, et.al., (2017) suggest that E-Spector provides detailed breakdowns of energy for each running process on the device, including applications running both

background and foreground services. In their study, Wang, et.al., (2017) evaluated and tested the overhead and accuracy of E-Spector and the results indicate that using E-Spector has the ability of providing an estimation of energy within a less than 10% error, as well as providing an estimation of energy overhead within a less than 4% error. However, tests energy model used by the authors only considers three hardware consumers of energy including; network (both cellular and WiFi network), the screen and the CPU, instead of considering all energy consumers thus presenting a key limitation to the study.

Moamen & Jamali (2015) are concerned with finding a solution that to sensor dependent applications that demand a lot of the phone's energy in order to continuously use sensor feed to provide services. The authors of the study believe applications that simultaneously monitor multiple sensors tend to amplify the problem as they consume significant amounts of the phone's battery (Moamen & Jamali, 2015). In their paper, Moamen & Jamali (2015) propose ShareSens as a potential solution to the above problem. ShareSens is an approach to merge applications' independent sensing requirements.

According to the authors of the report, this is achieved through utilising sensing schedulers for the sensors that would essentially determine the underlying lowest sensing rate which would mainly satisfy all the existing requests (Moamen & Jamali, 2015). Custom filters are then used to only send out the required data to each application on the device. Based on the report, any sensing requests that are made through the authors' proposed ShareSens API are generally sent to the respective schedulers that determine the overall optimum rates for sensing in order to satisfy all the prevailing requests.

Based on the experimental tests carried out on the ShareSens' capabilities, the authors found that there is significant power savings that can be attained when the ShareSens solution is used particularly when overlapping sensing requests exist (Moamen & Jamali, 2015). However, the current form of the ShareSens approach does not allow programmers to opportunistically choose sampling rates that are higher once they available, at a relatively low marginal cost.

In their study paper, Min, et.al., (2015) address the various factors that significantly impact phone batteries to the point of making their existing battery models become outdated and they further explore the initial approach that was aimed at helping phone users to understand the underlying cause and effect between the life of a phone's battery their physical activity. Min, et.al., (2015) proposed Sandra, a battery information adviser for smartphones that is designed to be mobility-aware. Sandra was designed with various key features including; a forecaster that provides estimates of battery life under different conditions of the user's future mobility, and an archive that is designed to provide past battery drain rates retrospective summary categorised by different conditions of mobility (Min, et al., 2015).

Based on the tests carried out the proposed approach, Sandra was found particularly helpful to smartphone users (Min, et al., 2015). However, the tool that Min, et.al., (2015) presented is neither an omniscient battery predictor nor a reconfiguration tool that extends batter's life like Power monitor v2. According to Min, et.al., (2015), Sandra's main goal is user enlightenment regarding new causal factors of their changes in mobility that impact the standby life of the phone batteries.

Besides choosing between network interfaces, the strength of the device signal has an influence on the consumption of the device's network. In their study, Schulman, et.al., (2010), proposed a scheduling algorithm that is designed to make use of a network signal with high strength. Their philosophy is that applications have to preferentially communicate when there is a strong network signal in order to realise energy savings, either through deferring communications that are not urgent or through advancing communications that are anticipated in order to coincide with strong signal periods (Schulman, et.al., 2010). To take advantage of a strong signal, Schulman, et al., (2010) developed a scheduling algorithm that focused on two specific kinds of applications, including streaming applications on one hand and sync applications on the other.

For streaming applications, the algorithm that the authors developed modulates the traffic stream in order to match with characteristics of radio energy while for sync applications the algorithm utilises flexible synchronisation intervals (Schulman, et.al., 2010). Their proposed energy-aware scheduling algorithm thereby takes into account tail energy as well as communication energy. Through their simulations and tests, Schulman, et.al., (2010) show that energy savings of up to 60% for on-demand streaming and up to 10% for synchronisation of email were attainable.

In a study conducted by, Zhang, et.al. (2010), the authors proposed the use of an online power estimation tool and a model generation framework in their contribution towards improving power-saving capabilities of Android smartphones on both the applications layer and the OS layer. Zhang, et.al. (2010) proposed a tool called the PowerTutor which was designed as an online power estimation system for the Android platform smartphones. The tool provides real-time, accurate power consumption estimates for components of the smartphone that are power intensive such as display, the CPU, cellular interfaces, GPS, and Wi-Fi interfaces (Zhang, et.al., 2010).

The PowerTutor was designed to be used by both application developers and smartphone users. Applications developers use to conveniently, accurately and rapidly determine the overall impact of changes in software design on power consumption while smartphone users can use the tool to determine the underlying power consumption characteristics the relate to competing mobile applications thus facilitating informed decision-making for both parties (Zhang, et.al., 2010). PowerTutor, according to Zhang, et.al. (2010) has a power model that includes six different components including: GPS, LCD display, CPU, audio interfaces, Wi-Fi and cellular interfaces. Based on the experiments that authors carried out, it was found that PowerTutor was accurate within an

average of 0.8% with at most 2.5% error for intervals of 10 seconds. In addition to the PowerTutor tool, Zhang, et.al. (2010) also proposed the PowerBooster tool which was designed an automatic state of battery discharge on the basis of a technique called the power model generation technique. According to Zhang, et.al. (2010), the experimental tests carried for 10-second intervals indicated that PowerBoost was accurate within 4.1%.

3.2 Monitor, detect and control app behaviour approach

Dao, et al., (2017) are concerned with the difficulty in identifying applications that are heavy power consumers on a smartphone as well as understanding why these applications are heavy power consumers. The authors believe that there is real need for phone users to be aware of applications on their smartphones that are heavy power consumers so that they are able to take appropriate action quickly enough be prevent their phone batteries being completely drained (Dao, et.al., 2017). In their study, Dao, et.al., (2017) propose TIDE, a tool that they believe can identify applications that are heavy energy consumers and provide an understanding of the reasons why an application is consuming a lot of energy on the phone. TIDE, according to Dao, et al., (2017) operates as user-centric tool which can be installed on a user's phone and it continuously the performed lightweight monitoring tasks on the application usage of the user as well as monitoring the resources that the application consumes.

Dao, et al., (2017) conduct an evaluation of their proposed tool using emulation of usage pattern traces from seventeen volunteer users and the results indicate that TIDE correctly estimated the energy consumption level for 225 applications out of 238. However, the tool does not provide a breakdown of the screen consumed energy in relation to individual applications yet the screen consumers the most amount battery power in most cases. Hence the results that the TIDE tool provide do not show the full picture of energy consumption.

Jabbarvand, et.al., (2015) were concerned with the fact that application repositories lack information regarding the relative energy cost of applications based on app categories which forces the user to install applications without appropriate understanding of the energy implications of these applications.

Wang, et.al., (2016) are concerned about the difficulty in the diagnosing energy inefficiency of applications that often use sensors to operate. In their study, Wang, et al., (2016) propose the GreenDroid approach that is designed to systematically diagnose problems associated with energy inefficiency among applications used in smartphones particularly those running on the Android platform. The proposed approach leverages the Application Execution Model (AEM) to realistically simulate the runtime behaviours of an application and it is also designed to have the ability of automatically analysing the sensory utilisation data of an application reporting the resulting

information to the application's developers (Wang, et al., 2016). Wang, et.al., (2016) evaluated the E-GreenDroid approach using 13 real applications on Android in two separate experiments and the results from the tests indicated that the tool was effective in executing its intended mandate. However, E-GreenDroid does not support concurrency of Android applications as it simply places all the execution into a single thread.

A solution presented in the existing literature that provides attempts to cover all areas of a smartphone's energy consumption is the Power monitor v2 that was proposed by Datta, Bonnet & Nikaein (2013). In their study, Datta, Bonnet & Nikaein (2013) suggest that the power monitor v2 is an Android application that works by employing a monitoring module to collect data which relates to all features of the smart device's (smartphone or tablet).

There are various modules, each collecting data on a specific feature including; the application monitor – collects data on running applications and their CPU load; battery monitor – collects data on battery status; CPU monitor – collects data on CPU operating frequency and load; the context monitor – collects data on system time, date and coarse location; the network monitor – collects data on the status of the mobile data, WiFi, network traffic used by applications and GPS status; and the display monitor – collects data on the screen timeout, level of brightness and devise interaction time (Datta, Bonnet & Nikaein, 2013). Based on their paper, Datta, Bonnet & Nikaein (2013) suggest the Power monitor v2 app monitors Android devices it is installed on continuously, stores the collected data locally for seven days and deploys a learning engine that is designed to generate various usage patterns that may exist within the smart device.

Thereafter power saving patterns for each pattern are generated dynamically. The collection of the usage data of the smart device raises various privacy related questions for the tool, however, Datta, Bonnet & Nikaein (2013) suggest that their approach preserves privacy of data since all the data collected stored and computations generated done locally. The evaluation tests carried out on the Power monitor v2 indicate that the application increased battery life of a Samsung GT-19100 running Android 2.3.4 OS by 8.2 hours while it increased the battery life of the Nexus 7 running Android 4.2.1 OS by 10 hours (Datta, Bonnet & Nikaein, 2013). Overall, the Power monitor v2 was found to increase the battery life of the devices it was installed on by 82% (Datta, Bonnet & Nikaein, 2013).

In their study, Dong & Zhong (2012) analysed the underlying influence of the content displayed on the overall energy-usage for displays whose design is based on the OLED technology. Through their research, the authors found that energy usage largely depends on the content displayed as different content contains different colours and for the devise to display different colours a certain amount of energy would be consumed (Dong & Zhong, 2012). Hence, Dong & Zhong (2012) concluded that designers of graphical user interface generally have a significant impact on the device's energy consumption. In this regard, Dong & Zhong (2012) proposed different energy

models which were designed to estimate the display content's power consumption. Dong & Zhong (2012) also proposed different transformation methods such as the utilisation of a lighter foreground colour and a dark background colour. Dong & Zhong (2012) used the transformation methods to evaluate the overall influence of their methods and found that energy usage can be reduced by approximately 75% hence saving the smartphone battery from draining.

Li, Tran & Halfond, (2014) used a similar idea to that presented by Dong & Zhong (2012) as they concentrated on the idea of reducing the consumption of energy by device-displays that use OLED technology. However, Li, Tran & Halfond, (2014) proposed a different approach in which they suggested that it is necessary to change the source code of the applications as a way of reducing the power consumption of the applications. They developed a tool they called Nyx which they suggested was capable of performing colour schemes transformations for applications (Li, Tran & Halfond, (2014). According to Li, Tran & Halfond, (2014), the test on their proposed solution found that battery savings of up to 40% for such modified applications were possible but only if users are willing to accept colour transformations in the name of saving battery.

Pathak, Hu & Zhang (2012) were concerned with the energy spent by mobile applications with the aim of finding ways to the reduce such energy consumption. In their study, Pathak, Hu & Zhang (2012) presented an energy profiler tool for Android smartphone applications called the Eprof. According their study, Eprof is an energy profiler that adopts the last-trigger accounting policy to capture intuitively the asynchronous modern smartphone components' power behaviour in mapping of energy activities to respective program smartphone entities (Pathak, Hu & Zhang, 2012). The tool was designed to be concerned with energy consumption profiling which is not linear as time and it has the capability of measuring intra-app consumption of energy including providing insights into the overall energy breakdown per application routine and per thread (Pathak, Hu & Zhang, 2012).

Their tool was also designed to be a general-purpose energy profiler that is fine grained works by assisting an application developer for Android smartphones to optimise the application's energy consumption. Pathak, Hu & Zhang (2012) carried out an experimental test which involved the profiling the energy consumption of six Android popular smartphone applications including; Facebook, Angry-Birds, and the Android Browser application among others. Their tests showed that Eprof shed light on the applications' internal energy dissipation and it further exposed surprising findings such as 65%-75% free applications' energy is consumed third-party advisement modules of the applications (Pathak, Hu & Zhang, 2012). Eprof also revealed numerous "wakelock bugs" (a family of smartphone applications energy bugs) and it efficiently pinpoints their location within the application's source code for to inform decision-making. Based on the experiments conducted by Pathak, Hu & Zhang (2012), their proposed accounting presentation of application 1/O energy (bundles) helped to reduce the consumption of energy of four applications involved in the test by 20% to 65%.

3.3 Switching off features approach

Petander (2009) proposed an energy-aware algorithm that was based on measurements of energy consumption in relation to 802.11 WLAN and UMTS networks on smartphones running on an Android operating system. The proposed algorithm generally utilises application traffic size estimations in order to determine the overall alternative of the minimum energy-cost through comparing the cost associated with the utilisation of UTMS with the underlying cost associated with performing a downward vertical opportunistic handoff back to WLAN, while utilising WLAN for data transfer (Petander, 2009). The authors show in their study that the proposed solution has the ability of predicting how much data will be transferred as a result of actions taken by the user. Based on experimental tests, Petander (2009) found that energy consumption of the smartphone increases by 18.3% whenever WiFi and UTMS are both powered on simultaneously, compared to powering on UTMS alone at any one time.

In their study, Taleb et al (2013) propose a technique that involves dynamic switching between WiFi and 3G communication on the smartphones. Taleb et al (2013) aim at achieving the ability to effectively switch to an alternative Wi-Fi connection from a primary cellular network. Taleb et al, (2013) conducted a set of experimental measures in relation to various network scenarios with the aim of identify the key components which affect consumption of energy within smart devices while they are connected to WiFi and 3G networks. The authors then used the measurement results to derive at generic analytical model for energy as a function of effective download bit rate and download data size (Taleb et al, 2013).

They developed an Android-based mobile application whose intended design is to test, in real scenarios, the overall performance of the algorithm for dynamic switching between WiFi and 3G connections. The results of the tests showed that it was possible to dynamically switch between WiFi and 3G communications and, when 3G only and WiFi only connections were compared, it was found that energy savings of 30% and 18% respectively were possible (Taleb et al, 2013). This particular study highlights the underlying potential benefits that intelligent switching within heterogeneous networks can provide.

In a study conducted by Cai et.al., (2015), the authors were focused on power wastage in mobile devices with 3G/4G networking that resulted from ‘tail time’ where the device’s radio is kept running despite the fact that no communication is taking place. Cai et.al., (2015) proposed DelayDroid as a framework which would provide a developer with the capability to add the required policies for reducing such energy wastage to existing Android application that are unmodified without any ‘human’ effort. The tool that Cai et.al., (2015) proposed uses bytecode refactoring and static analysis in order to identify method calls which send network related requests and modify the calls in order to detour them to the run-time of the DelayDroid.

The tool's runtime then batches them by applying a pre-defined policy, hence avoiding energy waste related to tail time hence improving energy efficiency. The universality and correctness of the DelayDroid mechanisms were evaluated and tested using 14 popular applications for Android and results indicated that DelayDroid was capable of reducing energy-waste related to 3G/4G tail time by 36% (Cai et.al., 2015). However, it is worth noting here that while the test results indicate that DelayDroid was effective in reducing the energy waste, it only reduces waste related 3G/4G tail time but not from screen and CPU usage which account for a large portion of the phone battery drain.

4. Conclusion

This report has provided a review of the existing literature regarding the different solutions, techniques and tools that have been proposed by different authors in response to battery energy consumption problems of mobile applications for smart devices running on the Android OS. The literature review covers studies that provide solutions based on three key approaches, including; approach 1 estimating and simulating power consumption of android applications, approach 2 monitoring, detecting and controlling the android applications' behavior, and approach 3 switching off smartphone features when not in use in order to reduce power consumption. Based on the review of the literature, solutions presented by prior studies in relation to approach 1 reveal that the average estimations that the proposed tools/techniques provide tend to conflict the actual usage habits of device and the accuracy of the power consumption measurements and simulators remains an issue of debate. The review of the existing literature in relation to the approach 2 reveals most solutions that monitor and control app behavior also consume power from the devise' battery for instance E-GreenDroid, Eprof, and among others.

Prior studies that propose solutions in the line of approach 3 reveal that the proposed techniques use predefined saving plans that provide a one-size-fits-all approach which does not necessarily provide customized/personalized solutions for users. Therefore, while the techniques presented herein provide some potential solutions for reducing energy consumption by mobile applications on Android-based smart-devices, they are limited in their usage.

References

Cai, H., Zhang, Y., Jin, Z., Liu, X. & Huang, G. 2015, "DelayDroid: Reducing Tail-Time Energy by Refactoring Android Apps", *Association for Computing Machinery - ACM*, pp. 1

Dao, T.A., Singh, I., Madhyastha, H.V., Krishnamurthy, S.V., Cao, G. & Mohapatra, P. 2017, "TIDE: A User-Centric Tool for Identifying Energy Hungry Applications on Smartphones", *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1459-1474

Datta, S.K., Bonnet, C. & Nikaein, N. 2013, "Power monitor v2: Novel power saving Android application", *The Institute of Electrical and Electronics Engineers - IEEE*, pp. 253

Dong, M. & Zhong, L. 2012, "Power Modeling and Optimization for OLED Displays", *IEEE Transactions on Mobile Computing*, vol. 11, no. 9, pp. 1587-1599

Li, D., Tran, A.H. & Halfond, W.G.J. 2014, "Making web applications more energy efficient for OLED smartphones", *Association for Computing Machinery - ACM*, pp. 527

Min, C., Yoo, C., Hwang, I., Kang, S., Lee, Y., Lee, S., Park, P., Lee, C., Choi, S. & Song, J. 2015, "Sandra helps you learn: the more you walk, the more battery your phone drains", *Association for Computing Machinery - ACM*, pp. 421

Moamen, A.A. & Jamali, N. 2015, "Share Sens: An Approach to Optimizing Energy Consumption of Continuous Mobile Sensing Workloads", *The Institute of Electrical and Electronics Engineers - IEEE*, pp. 89

Pathak, A., Hu, Y.C. & Zhang, M. 2012, "Where is the energy spent inside my app? fine grained energy accounting on smartphones with Eprof", *Association for Computing Machinery - ACM*, pp. 29

Petander, H. 2009, "Energy-aware network selection using traffic estimation", *Association for Computing Machinery - ACM*, pp. 55-60

Schulman, A., Navda, V., Ramjee, R., Spring, N., Deshpande, P., Grunewald, C., Jain, K. & Padmanabhan, V. 2010, "Bartendr: a practical approach to energy-aware cellular data scheduling", *Association for Computing Machinery (ACM)*, pp. 85

Taleb, S., Dia, M., Farhat, J., Dawy, Z. & Hajj, H. 2013, "On the Design of Energy-Aware 3G/WiFi Heterogeneous Networks under Realistic Conditions", *The Institute of Electrical and Electronics Engineers - IEEE*, pp. 523

Wang, C., Guo, Y., Shen, P. & Chen, X. 2017, "E-Spector: Online energy inspection for Android applications", *The Institute of Electrical and Electronics Engineers - IEEE*, pp. 1

Wang, J., Liu, Y., Xu, C., Ma, X. & Lu, J. 2016, "E-greenDroid: effective energy inefficiency analysis for android applications", *Association for Computing Machinery - ACM*, pp. 71

Westfield, B. & Gopalan, A. 2016, "Orka: A new technique to profile the energy usage of Android applications", *SciTePress*, pp. 1

Zhang, L., Tiwana, B., Qian, Z., Wang, Z., Dick, R., Mao, Z. & Yang, L. 2010, "Accurate online power estimation and automatic battery behavior-based power model generation for smartphones", *Association for Computing Machinery - ACM*, pp. 105