International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 9, No.4 (July-2020)

http://dx.doi.org/10.12785/ijcds/090419

S3: A Study on the Efficiency of Current Power-Saving Approaches Used Among Android-Application Development & Usage Stages

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

[1] College of Computer and Information Sciences, Prince Sultan University, Riyadh, Saudi Arabia
[2] Science and Technology Faculty, University Fernando Pessoa, Porto, Portugal

Received 03 Feb. 2020 Revised 2 Apr. 2020, Accepted 01 May. 2020 Published 1 Jul. 2020

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 sub-field 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 paper 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: Energy, Power, Smartphones, Android, Apps, Saving, Consumption, Preventive, Rating

1. Introduction & Backgrond

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 [11].

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 [7].

This paper reports different research themes towards the reduction of smartphone power consumption. We present two different avenues to prolong the batter life of Android devices. The first avenue includes application power rating through a 6-stars score components- usage pattern analysis to generate power saving ratings (preventive). Then efforts have been made to survey Android power saving apps available in Google play Apps store as the basis to find out different power saving detective approaches, operations and limitations. The usage pattern-based power saving profile generation during runtime (detective) has several powerful features that depend on monitoring the actual applications during running and reporting abusage.

The structure of this paper goes as follows: section 2 identifies the main current power-saving approaches and presents an average lifecycle that shows the main stages of an average android application with the presence of each Estimate and simulate power consumption

A. *Review of Concept Implementations*

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

Email: aalmasri@psu.edu.sa, lmbg@ufp.edu.pt

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 [14]. 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. 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. 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 utilized. 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 & Gopala, Okra 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 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, 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, propose E-Spector as a potential online based tool/method the inspects energy usage, visualizes 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, 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 recognize what actually took place behind each spike in energy usage [12][15]. 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. Furthermore, Wang, 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, 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 Wi-Fi network), the screen and the CPU, instead of considering all energy consumers thus presenting a key limitation to the study.

Moamen & Jamali 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 [12]. In their paper, Moamen & Jamali 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 utilizing sensing schedulers for the sensors that would essentially determine the underlying lowest sensing rate which would mainly satisfy all the existing requests. 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. 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. Power-saving approach among the lifecycle stages. Then sections 3 through 5 present the detailed review of each power-saving approach. We then conclude in Section 6.

2. METHODOLOGHY

In order to demonstrate the main issues with current Power-Saving approaches, first we proposed summarizing the current power saving approaches that are used in today's smartphones in a S3 classification (Simulate, Supervise and Sacrifice), the following classification were made: Approach 1 reflects the concept of "simulating and estimating" the energy consumption of android apps before making these apps available to end users by implementing green coding techniques, energy-aware designs, mobile battery simulators, historical analytical data .. etc. The predominant purpose of this approach is to direct android app designers and developers by considering metrics and measurements created by simulators and profilers of power consumption in addition to green code readers and evaluators. It also aims towards detecting energy leaks and measuring overall power consumption based on multi factor models which may include: memory usage per minute, processing time, background live services, loopholes that may cause continues unintended running and length of code. This approach also aims at reducing or eliminating the use of the platform's local processing and storage resources by offering a web link under a simple device interface (Thin-Client and/or Cloud). An example of a real-life application of this approach is the power consumption profilers/compilers and green code readers/evaluators e.g. PowerTutor [15] which was developed for Android platform smartphones as an online power estimation system. Also, the energy-aware-best practices suggested on the official android developer's website [18] e.g. "Remove location updates", "Set timeouts" in addition to all cloud-based-processing apps which relies only on the internet by providing a link or a web-based interface to users.

Approach 2, follows the "Supervise, detect and control" philosophy, so it applies this on the behavior of an android application while it is running on an end-user's phone and optimizing the power consumption. Its basic concept is to run a real-time power-usage monitoring code and then keep notifying the user by proposing instant actions to optimize power usage. It also takes the chance whenever needed to interrupt the built-in runtime schedule of background apps in order to kill whatever apps it may found causing a high level of power consumption, it also applies the same for features called by applications e.g. continuous synchronization. The most common example of a real-life implementation of this approach is the battery optimizers available in various app stores, commonly known as "battery-saving apps," along with built-in power management and saving algorithms originally equipped with smartphones.

Approach 3, is more about "Sacrifice" smartphones technology or performance by switching off a number of features for the sake of saving power philosophy. Its basic concept is to sacrifice smartphone's modern features for the sake of saving power by switching some of the optional modern features off or by reducing the level of the smartphone's performance or any of its components. An example of a real-life application of this approach is the "power-saving modes" that are used among different brands, models and versions of android smartphones e.g. Samsung Maximum Power Saving Mode [19].

3. SIMULATE AND ESTIMATE

A. Review of Concept Implementations

In their paper, Joakim v. Kistowski, Maximilian Deffner and Samuel Kounev introduced a model to predict the power consumption of component placements at run-time based on the load and power profile collected for a running distributed application in a heterogeneous environment. They were able to predict the amount of consumption based on load intensity and performance counters with percentage error of 2.21% and with an error of 1.04% when predicting a previously unobserved load level [17]. Min C, from a similar perspective, discussed the various factors that have a significant impact on telephone batteries to the point of making their current battery models obsolete and further examined the initial approach aimed at helping telephone users understand the underlying cause and effect of their physical activity between the battery life of a smartphone. Min C, suggested Sandra, a mobility-aware battery information advisor for smartphones [6]. Sandra was developed with various key features including; a forecaster that offers forecasts of battery life under different conditions of potential mobility of the user, and a database designed to provide a historical overview of past battery drain levels, classified by different mobility conditions. Sandra was found to be particularly helpful to mobile users, based on the tests performed by the proposed solution.

TABLE I. ARTICLES IMPLEMENTING EACH APPROACH

Implementing the SIMULATE & ESTIMATE Approach
Min, C., Yoo, C., Hwang, I., Kang, S., Lee, Y., Lee, S., Park, P., Lee, C., Choi, S. & Song, J. 2015. [6]
J. V. Kistowski, M. Deffner, and S. Kounev, 2018 [17]
F. Uddin, 2017 [10]
Wang, C., Guo, Y., Shen, P. & Chen, X. 2017 [12]
Westfield, B. & Gopalan, A. 2016 [14]
Zhang, L., Tiwana, B., Qian, Z., Wang, Z., Dick, R., Mao, Z. & Yang, L. 2010 [15]
Implementing the SUPERVISE, DETECT & CONTROL Approach
Dao, T.A., Singh, I., Madhyastha, H.V., Krishnamurthy, S.V., Cao, G. & Mohapatra, P. 2017 [2].
Wang, Y. Guo, P. Shen, and X. Chen, "E-Spector: Online energy inspection for Android applications," 2017 [3].
F. Chesterman, G. Muliuk, B. Piepers, T. Kimpe, P. D. Visschere, and K. Neyts, 2016 [4].
Li, D., Tran, A.H. & Halfond, W.G.J. 2014 [5]
A. Banerjee and A. Roychoudhury, 2016 [8]
Wang, J., Liu, Y., Xu, C., Ma, X. & Lu, J. 2016 [13]
Pandey, N., Verma, O. and Kumar, A., 2019 [20]
Implementing the SACRIFICE MODERN FEATURES Approach
Cai, H., Zhang, Y., Jin, Z., Liu, X. & Huang, G. 2015 [1]
M. F. Tuysuz and M. Ucan, 2017 [9]
Taleb, S., Dia, M., Farhat, J., Dawy, Z. & Hajj, H. 2013 [11]
Vishwakarma, S. and Bhadauria, S., 2015 [21]
Alshurafa, Nabil & Eastwood, JoAnn & Nyamathi, Suneil & Xu, Wenyao & Liu, Jason & Sarrafzadeh, Majid. (2014). [22]

However, the tool that Min, presented is neither an omniscient battery predictor nor a reconfiguration tool that extends batter's life like Power monitor v2. According to Min, Sandra's main goal is user enlightenment regarding new causal factors of their changes in mobility that impacts 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, Forkan Uddin, proposed a scheduling algorithm that is designed to make use of a network signal with high strength. His philosophy is that applications have to preferentially communicate when there is a strong network signal in order to realize 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 [10]. To take advantage of a strong signal, Forkan Uddin, 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 utilizes flexible synchronization intervals. His proposed energy-aware scheduling algorithm thereby takes into account tail energy as well as communication energy.

In a study conducted by, Zhang 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 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 [15]. 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. PowerTutor, according to Zhang, 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 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, the experimental tests carried for 10-second intervals indicated that PowerBoost was accurate within 4.1%.

B. Limitations

From the previous implementations of the main concept thy all share the same issue which is addressed to the approach. The first issue is that not all hardware is created equal and not all apps works the same on all hardware. In the same time, methods of normalizing energy consumption measurements across different platform with either linear or nonlinear scaling still ignores the usage habits and the lifestyle of the end user (Heavy gamer, Outdoor Field Engineer) Figure 1 shows workload deployment power prediction data flow model proposed by Joakim v. Kistowski, Maximilian Deffner and Samuel Kounev, it is clear that the model does not include any of the user characteristics which may influence the final result. Added to this, apps energy consumption is not stable across different versions of the app. Furthermore, testing a single build of the code might not be enough, a partial or entire energy consumption profile should probably be built. In addition to all the previous, continuous need of internet connectivity uses local connection hardware resources that require complicated power aware connectivity algorithms on both the sender and receiver side (router, tower, satellite.. etc) in addition to the connection method and technology. Finally, downtime is often cited as one of the biggest disadvantages of cloud computing. Since cloud computing systems are internet-based, service outages are always an unfortunate possibility and can occur for any reason, which is the biggest challenge that faces most thin-client solutions nowadays.

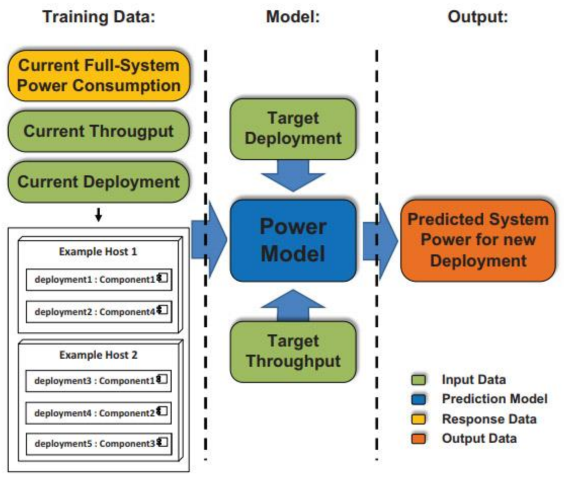

Figure 1. Workload deployment power prediction data flow with example deployment [17].

4. Supervise, Detect and Control App Behavior

A. Review of Concept Implementations

Dao, 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 preventing their phone batteries being completely drained [2]. In their study, Dao, proposes 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, operates as user-centric tool which can be installed on a user's phone and it continuously performs lightweight monitoring tasks on the application usage of the user as well as monitoring the resources that the application consumes. Dao, 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 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 is concerned about the difficulty in the diagnosing energy inefficiency of applications that often use sensors to operate. In their study, Wang proposes 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 behaviors of an application and it is also designed to have the ability of automatically analyzing the sensory utilization data of an application reporting the resulting information to the application's developers [13]. Wang 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 E-Spector that was proposed by Chengke Wang, Yao Guo, Peng Shen and Xiangqun Chen. In their study they suggest that the E-Spector 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, Wi-Fi, 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 [3]. They suggest that E-Spector, stores the collected data locally 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.

In their study, F. Chesterman, G. Muliuk, B. Piepers, T. Kimpe, P. D. Visschere, and K. Neyts analyzed 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 colors and for the devise to display different colors a certain amount of energy would be consumed [4]. Hence, the authors concluded that designers of graphical user interface generally have a significant impact on the device's energy consumption. In this regard, they proposed different energy models which were designed to estimate the display content's power consumption. The authors also proposed different transformation methods such as the utilization of a lighter foreground color and a dark background color. They also 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. In their study Pandey, N., Verma, O. and Kumar proposes NIPO, a framework which aim to optimize smartphone power consumption [20]. The NIPO framework is generic in nature in that it can be used for power optimization on any smart device or computer subsystem. It has system-based client-server architecture. Many of its components reside on the server while others are on the device. The main components of the proposed system on the device side are: User Interface module, Battery Monitoring module, Power State Monitoring module, Time Monitoring module and Server Interface module. On the server side, the two main components of the system are: Device Interface module and Power Optimization Engine. The key task was to select the optimum parameters (Brightness and Transparency) by maintaining a defined battery level to fulfill the time-to-last requirements. Their analysis found that the till battery power level is above 30 percent, by controlling transparency and brightness, the battery power was optimized. Adding brightness to the power optimization variables therefore allows improved power control. However, NIPO does not endorse Android applications participation, as it simply places all execution at parameter-based monitoring stage.

Li, Tran & Halfond, used a similar idea to that presented by Dong & Zhong as they concentrated on the idea of reducing the consumption of energy by device-displays that use OLED technology. However, Li, Tran & Halfond 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 [5]. According to Li, Tran & Halfond, 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. Abhijeet Banerjee & Abhik Roychoudhury were concerned with the energy spent by within an application with the aim of finding ways to reduce such energy consumption. In their study, the authors presented a design-expression that can be described as a regular-expression representing the ordering of energy-intensive resource usages and invocation of key functionalities (event-handlers) within the app. According to their study, they used a refactoring tool that adopts the last-trigger accounting policy to capture intuitively the asynchronous modern smartphone components' power behavior in mapping of energy activities to respective program smartphone entities [8]. 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. 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 optimize the application's energy consumption. Their tests showed that refactoring 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. They 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 Abhijeet Banerjee & Abhik Roychoudhury, 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 3 % to 29 %.

B. Limitations

The main and critical issue of the concept which is addressed by its implementations is that is requires power which causes it to fail in delivering its main goal of saving energy. Simply, monitoring and announcing consumes power for the sake of saving power. Plus, whatever runs on the application and/or the OS layers of the phone consumes power from the same phone battery. By using any power usage profilers in order to rank the usage of power consumed by each running application we were able to rank two power-optimizing applications both as one of the top 10 most power-consuming applications as Figure 2 shows.

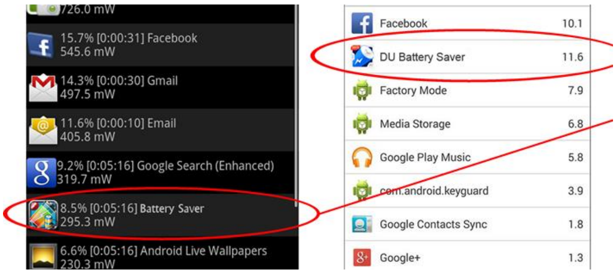

Figure 2. Using PowerTutor and Trepn Profiler to read the amount of energy consumed by two popular power-optimizing applications.

5. SACRIFICE MODERN FEATURES

A. Review of Concept Implementations

Mehmet Fatih Tuysuz & Murat Ucan 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 utilizes application traffic size estimations in order to determine the overall alternative of the minimum energy-cost through comparing the cost associated with the utilization of UTMS with the underlying cost associated with performing a downward vertical opportunistic handoff back to WLAN, while utilizing WLAN for data transfer [9]. 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 found that energy consumption of the smartphone increases by 18.3% whenever Wi-Fi and UTMS are both powered on simultaneously, compared to powering on UTMS alone at any one time.

In their study, Taleb proposes a technique that involves dynamic switching between Wi-Fi and 3G communication on the smartphones. Taleb aims at achieving the ability to effectively switch to an alternative Wi-Fi connection from a primary cellular network. Taleb 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 Wi-Fi 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 [11]. 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 Wi-Fi and 3G connections. The results of the tests showed that it was possible to dynamically switch between Wi-Fi and 3G communications and, when 3G only and Wi-Fi only connections were compared, it was found that energy savings of 30% and 18% respectively were possible. This particular study highlights the underlying potential benefits that intelligent switching within heterogeneous networks can provide. A method proposed by Vishwakarma, S. and Bhadauria was to create a cooperative network where users with high battery level support bring low battery level user traffic [21]. This scheme helps to increase the amount of valued battery in the network, thereby reducing the probability that users run out of battery early. The technique takes the form of a service that uses a device-to-device communication architecture instead of using direct network connection e.g. 3G or 4G. By reducing the power consumed by communications, the proposed system provided considerable gain for the overall battery life. However, the proposed solution relies on the existence and usage of the hexagonal cell environment, *helpees* and D2D communication which also raises concerns related to data privacy and security.

In a study conducted by Cai 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 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 'huhman' effort. The tool that Cai 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% [1]. 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.

WANDA-CVD was introduced by authors in [22] with a new method of battery optimization that improves the battery life of smartphones tracking physical activity. Their technique of battery optimization increased the battery life by 300%. The smartphone could enter sleep mode to decrease the accelerometer's sampling rate while the user is not in motion, knowing that individuals spend most of their day inactive. Authors created a battery management technique such that it reaches an initial state when the phone is connected to a charger, where the accelerometer can be switched off. It enters an active state once the phone is disconnected, where the accelerometer is switched on and the sampling rate is set at 10Hz. If the user is idle, sits on a sofa or dinner table, no physical movement is detected and the device enters an inactive state where the sampling rate drops to 1Hz. In both in-lab and real-world environments, the WANDA-CVD device has proven effective in battery optimization. The healthcare industry could benefit from such a system. However, active auto-synchronization is considered as an essential requirement for most smartphones' users. So, the usage of sleep modes for the sake of saving energy is to be reconsidered as a current solution.

B. *Limitations*

From the previous implementations the main issues of the concept are related to the predefined saving plans which does not differentiate between smartphone users in terms of using habits and claims to provide a one-size-fits-all solution which converts the colorful screen of the smartphone to a semi-black and white old screen for either a heavy gamer or a 70 years old user as shown in Figure3. Additionally, Flexible Saving Plans relies on user's personal estimations to control the phone components/technologies "on/off switches" which the user may never bother to go through. Furthermore, the idea of having a "One Size Fit All" power saving plans is gradually receding [16]. Also, these power-saving modes does not serve different users categories based on their usage of different applications categories which causes the majority of users to avoid using these power saving modes for reasons related the user's need to use the new technology itself rather than sacrificing it for the sake of saving energy. Figure 4 shows the result of a simple survey that we implemented on a sample of ten users from six users' categories which either care about the technology or the battery life.

6. Discussion and Future Research Direction

We believe that we have implemented a deep review study that derived us to a set of results which can support proposing a new power saving approach to be used in smartphones. Our main contribution is to create a new research direction towards helping in proposing a new power saving approach that does not require consuming any of the energy of the same battery while keeping the user able to both save energy and enjoy all of the new features of a smartphone. Although there are various improvements in the area of smartphones applications development in terms of power saving, the inclusion of many other factors which are mainly related to using habits of smartphones users may cause for example pop up advertisements and other interrelated factors which force the smartphone user to consume energy from the battery. As per to the battery saving features that could help in resistance of battery includes the flight mode, do not disturb, powering off Wi-Fi and data connections. Moreover, with respect to the research questions regarding the fact that power saving application consumes a lot of energy it is clearly stated by the past studies that due to many factors there is a lot of battery exploitation and these issues have been becoming a major issue for smartphone users and manufactures. The study provides a clear insight that there are several ways which could be designed and developed to save the power of smartphones more efficiently.

While much attention has been drawn to energy saving, there are many challenges that the current researchers need to pay more attention to. In this study, a general investigation of these battery savings is presented. The study tackled these battery savings implemented on the smartphone's applications layer, OS layer and even during the development stage of an android app. A primary motivation for future research is to improve end-users 'involvement in exploiting the degree of versatility between the aforementioned S3 approaches while preserving or improving energy consumption levels. A good area of future research to resolve this particular common limitation is to find a solution that would not require end-users to compromise all or parts of a technology that is considered "required" while also allowing end-users to extend the battery life of their phone. Another future research area is to avoid adopting the argument that "waste some energy from the battery of a smartphone under the pretext of saving the same battery's energy. Finally, additional efforts are to be made to define the fundamental shortcomings of the battery savings that are developed and implemented in future smartphones.

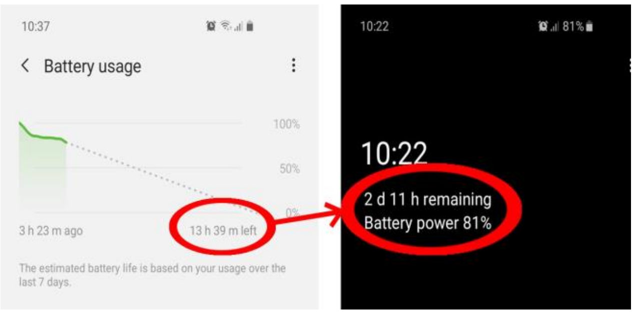

Figure 3. Activating "Ultra/Maximum Power Saving Mode" Gives more battery life time at the expense of technology (Less or No Colors).

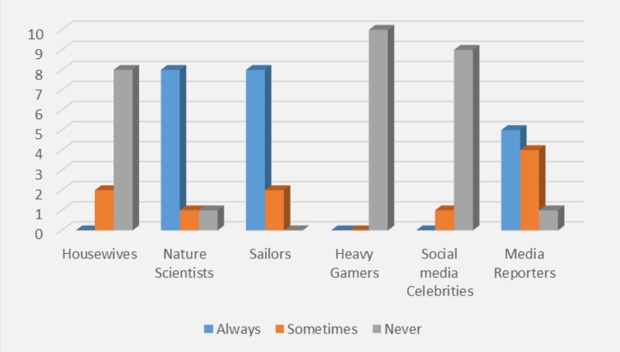

Figure 4. Usage of "Ultra Power Saving Modes" among six categories os smartphones users..

7. Conclusion

This paper 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.

Acknowledgments

The authors graciously thank the College of Computer and Information Sciences, Prince Sultan University and the Faculty of Science and Technology, University Fernando Pessoa for their scientific and administrative support.

References

[1] 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

[2] 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

[3] Wang, Y. Guo, P. Shen, and X. Chen, "E-Spector: Online energy inspection for Android applications," 2017 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED), 2017.

[4] F. Chesterman, G. Muliuk, B. Piepers, T. Kimpe, P. D. Visschere, and K. Neyts, "Power Consumption and Temperature Distribution in WRGB Active-Matrix OLED Displays," Journal of Display Technology, vol. 12, no. 6, pp. 616–625, 2016.

[5] 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

[6] 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

[7] 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

[8] A. Banerjee and A. Roychoudhury, "Automated re-factoring of Android apps to enhance energy-efficiency," Proceedings of the International Workshop on Mobile Software Engineering and Systems - MOBILESoft 16, 2016.

[9] M. F. Tuysuz and M. Ucan, "Energy-aware access point selection for smartphones," 2017 25th Signal Processing and Communications Applications Conference (SIU), 2017.

[10] F. Uddin, "Energy-Aware Optimal Data Aggregation in Smart Grid Wireless Communication Networks," IEEE Transactions on Green Communications and Networking, vol. 1, no. 3, pp. 358–371, 2017.

[11] 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

[12] 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

[13] 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

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

[15] 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

[16] A, Almasri, Preventive Customized Strategies for Saving Energy on Smartphones, M.Sc. Thesis, Prince Sultan University, 2015.

[17] J. V. Kistowski, M. Deffner, and S. Kounev, "Run-Time Prediction of Power Consumption for Component Deployments," 2018 IEEE International Conference on Autonomic Computing (ICAC), 2018

[18] Android Developers, Android Developers. 2020. Available online: https://developer.android.com/ (accessed on 16 March 2020).

[19] Samsung Electronics America. 2020. Use Power Saving Modes On Your Galaxy Phone. [online] Available at: <https://www.samsung.com/us/support/answer/ANS00079037/> [Accessed 19 March 2020].

[20] Pandey, N., Verma, O. and Kumar, A., 2019. Nature Inspired Power Optimization in smartphones. Swarm and Evolutionary Computation, 44, pp.470-479.

[21] Vishwakarma, S. and Bhadauria, S., 2015. Smartphone Battery Optimization by Reducing Energy Consumption in Network Communications. International Journal of Engineering Research and General Science, 3(2), pp.208-217.

[22] Alshurafa, Nabil & Eastwood, JoAnn & Nyamathi, Suneil & Xu, Wenyao & Liu, Jason & Sarrafzadeh, Majid. (2014). Battery optimization in smartphones for remote health monitoring systems to enhance user adherence. ACM International Conference Proceeding Series. 2014. 1-4. 10.1145/2674396.2674407.

Abdullah Mahmoud Almasri holds a Bachelor of science degree in Information Systems and a Master of science degree in Software Engineering from Prince Sultan University, Saudi Arabia. Senior lecturer and researcher of Computer Science and Information Systems at Prince Sultan University since 2015. His current research interests include Mobile Computing, Power Aware Applications and Systems Design.

Luis Borges Gouveia holds a Habilitation in Engineering and Industrial Management from University of Aveiro, Portugal and a PhD in Computer Science from University of Lancaster, UK. Full Professor at University Fernando Pessoa, Porto, Portugal where is the coordinator of the PhD program in Information Science. His main interests are related with how computers and digital applications can impact people, and include novel application to provide better sustainable technology usage.