Survey on adaptation techniques of energy consumption within a smartphone

Khalil Ibrahim Hamzaoui, Gilles Grimaud, Mohammed Berrajaa, Mostafa Azizi, Abdelkader Betari

Survey on adaptation techniques of energy consumption within a smartphone

Khalil Ibrahim HAMZAOUI
IRCICA 2XS/ LANOL

Lille,France/ Oujda,Maroc

hamzaoui.khalil@gmail.com

Gilles GRIMAUD
IRCICA 2XS

Lille,France

gilles.grimaud@lifl.fr

Mohammed BERRAJAA
LANOL

Oujda,Maroc

berrajaamo@yahoo.fr

Mostafa AZIZI
MATSI

Oujda,Maroc

azizi.mos@gmail.com

Abdelkader BETARI
MATSI

Oujda,Maroc

betari.ump@gmail.com

***Abstract*—The energy consumption into a smartphone is defined by the energy cost necessary for the components equipment to achieve their activities. This activity is induced by software executions related to users' activity. Indeed, the software produces during a given time an amount of work, (e.g. I/O access, and data encoding/decoding), that grows the number of operations over inner equipment. In other words, the energy consumption could results from the execution of the different interactions between hardware, software and users which by their behaviors trigger a workload on hardware components. The assessment and measurements of the cost of consumed energy, as well as problems in the methods used for energy optimization, are the main topic of this investigation. As results of this work, we conclude that some of the investigated techniques are more accurate than the others for tracking the main sources or equipment responsible of consuming energy.**

Keywords : Smartphone; Mobile, Operating system , EPROF ; AppScope ; Cinder ; TailEnder ; RAPL ; Energy ; consumption;

1. INTRODUCTION

Despite the success of smartphones on the market and the exponential growth of mobile applications, their usefulness has been and will be severely limited by the life or autonomy of their batteries. That is why the optimization of energy consumption by smartphone applications is of critical importance. However, most of mobile applications developed so far have been designed unconsciously of their consumption of energy [14].

The energy consumption of a smartphone is defined by the amount of the used energy to perform its services. The gap between the energy stored in the battery and the energy consumed by the main components is increasing with each new generation of smartphones. The energy consumption results from a smartphone consumption of its components involved in running its applications. It is therefore important to measure and understand how the energy is consumed in these mobile devices.

The terminals are supplied by the battery to allow the highest degree of freedom to the user; but it limits resources in terms of energy and power. It is essential to understand the difference between these two terms that are sometimes used interchangeably [1].

In addition, a component of o smartphone can have one on more levels of power state:

The active state : The application processor is operational.

The inactive state : the application processor is inactive but the communication processor generate a low level of activity.

The state of tail : The device is not in the inactive state but no application is activated.

The research presented in this article is part of the development of models. That enables the modeling and evaluation of energy cost in mobile environments, and more specifically in the Android platforms as well as studies on the energy behavior of the different components of smartphones.

The second section will focus on the evaluation, the importance as well as the measurement of energy cost.

We conduct our survey on the following models: Tailender, Cinder, Eprof, RAPL and AppScope.

In the third section, we will begin by the analysis and comparison of the models mentioned above. Then, in the remaining sections, we will make a synthesis of different methods and try to get them classified.

2. EVALUATING THE ENERGY COST

Energy on mobile phones is a valuable resource. As all mobile phones are equipped with multiple wireless technologies such as 3G, GSM and Wi-Fi, it is important to understand the characteristics related to energy consumption.

2.1. Importance of accurate assessment of energy costs

In operation of each service in a smartphone there is energy on an amount of energy dissipated. It is important to know how to measure and understand how energy is consumed on mobile devices to design solutions that reduce energy consumption in order to improve performance and user experience.

Modern smartphones are equipped with a large variety of integrated circuits. These include among others the CPU [7], memory, SD card, Wi-Fi, telephone, Bluetooth, GPS, camera, accelerometer, digital compass, LCD [8] screen or touch screen, microphone, speaker... It is common for applications to use smartphones more components then necessary to provide a richer user experience.

Knowledge of the energy consumption of the different components of the smartphone is the pivot axis to know the essential parts that consume more energy. these parts will be prioritized to optimize the energy cost. This step is very important because any misclassification will prevent finding an optimal solution.

2.2. Measurement of energy cost

At the time of each service operates in a smartphone, an energy consumption of a quantity of energy dissipated. In this article, we will conduct our survey on the following techniques: Tailender, Cinder, Eprof, RAPL and AppScope. These models can show the main sources of energy for the purpose of optimization of energy reserved for different devices.

We briefly describe the principle of implementations and limitations of each method listed.

A. TailEnder

In 2009, Niranjan Balasubramanian, Aruna Balasubramanian and Arun Venkataramani have conceived TailEnder [2] to measure the energy consumption characteristics of different equipment smartphones.

The purpose of TailEnder is the minimization of energy consumption for applications that can tolerate a small delay such as e-mails. In particular, the study of the characteristics of power consumption of 3G reveals significant and non-intuitive for the design of effective implementation of the energy implications. Analytical modeling: previous work such as [3, 4, 5] studied the impact of different energy saving techniques in 3G networks using analytical models.

Based on this finding, a simple model of energy consumption has been developed in order to identify opportunities for reducing the energy of the network induced by common network applications for each activity of the three consumption technologies mentioned above.

The experiments show that the protocol TailEnder gives the ability to download 60% more and make 50% more web search compared to the same consumption with a default policy [2].

TailEnder is evaluated using simulation-based experiences on the phone model; experiments were conducted based on Nokia N95.

The object of the evaluation is to quantify the reduction in energy use when using TailEnder for different applications, based on a protocol by default. To show the general applicability of TailEnder, its performance is evaluated on the following applications [2]: e-mails, news feeds and Web search. E-mail and news feeds are applications that can tolerate a moderate delay, Web search is an interactive application, but can benefit from prefetching.

The impact of TailEnder for energy minimization depends largely on the application traffic and user behavior.

If a user receives an e-mail every hour, or if new feeds are updated once per hour, the probability that TailEnder brings the benefits of energy management is minimal.

B. Cinder

In 2011 A. Roy, SM Rumble, R. Stutsman, P. Levis, D. Mazieres, and N. Zeldovich show in [6] that Cinder method can model the energy consumption and is suitable for partitioning applications to energy terminals, even with complex policies.

Cinder is designed for mobile phones and devices, it allows users and applications to control and manage the limited device resources such as energy. This mechanism, in contrast to previous approaches, introduces two new abstractions that aim to identify the primary responsible for resource consumption.

The priority task is visualizing the problem and assigns applications to consumption guidelines. [7, 8, 9, 10]. The principle of Cinder [6] is to have information on three mechanisms: isolation, subdivision and delegation.

The insulation overlooked the processor ensures that the process will not die of hunger. In this part we interesting in isolation from consumption.

For example two processes P1 and P2 each consume so much energy should not have access to all resources (even through the son or pipe).

The power reserved to the emergency call must be isolated so in no way a program can be used [6]

- Cinder is based on HISTAR core, an exokernel;

- Cinder uses the concept of reserves and taps.

A reserve describes the right of a given resource such as energy; it prevents the execution of applications that do not have sufficient resources quantity.

This method is effective to strangle energy consumption. In theory, the reserves are sufficient to control the system in terms of resource use, but this approach may be ineffective in practice.

The taps' concept can remedy this problem by transferring resources that could be implemented by special purpose son moving explicitly between resources and reserves.

The first reserve is connected to the battery by a high speed valve. The second is a low flow reserve energy connected to the battery through a low-flow tap.

The Figure1[6] explains the mechanism by setting an example of a web browser set to run for at least six hours on a battery 15kJ

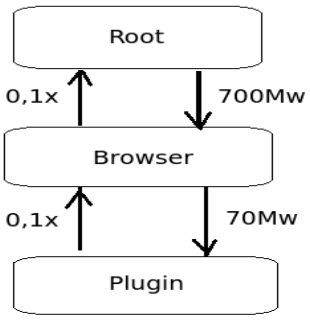

Figure1: Root reserve,connected to a reserve via a tap. The battery is protected from being misused by the web browser [6]

The browser also ensures that the Plug-in cannot use more than 10% of its énergie.0.1x (x being the total energy); proportional taps prevent the browser and the Plug-in hoarding energy [6].

With Cinder, the user can make a quota for web browsing (reserve and tap on particular page and Plug-in) in order to ensure the implementation of the Plug-in while being sure that it not die of hunger.

For each application, Cinder divides the reserve in two other reserves through the taps.

When an emergency call is initiated, the device closes the valve on all other running applications (0mW), they will not have access to resources until the emergency call is present.

Figure 2 shows how to keep energy value (250 J) set next to an emergency call:

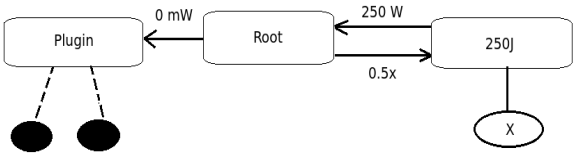

Figure 2: Maintain the value of energy for emergency call

When an emergency call is detected, the supply valve for normal applications is defined at 0mW. The proportional valve prevents the task in question (X task) the capitalization of energy during normal operation.

Cinder can control the energy by the subdivision, delegation and isolation; he also provides visibility into the energy and power of a system running.

Examining how applications use the data path of the phone, we estimate that Cinder can improve the energy efficiency of a system of management of complex systems with nonlinear energy consumption.

We note that Cinder simplifies the use of policies that enable efficient use of expensive devices. However, it presents some limitations:

- Cinder is designed in a professional environment (it uses units of measurement such as Watt and Joule) which is inadequate for the end user / developer who have not the ability to interpret these values (duration or quantity remaining to download …)
- Cinder and Android are developed under Linux, As Android features and applications are based on the Dalvik virtual machine, Cinder must also be suited with Dalvik so it can support all Android applications.
- Cinder is not able to take the decision on the ability to consume through resource (tasks to be performed by users and applications), so the allocation of resources is not necessarily optimal.
- The competition control and access security must be better managed: storage of privileges in taps must be secure and must allow taps to transfer resources between reserves.

C. Eprof

In 2012 Abhinav Pathak, Y. Charlie Hu and Ming Zhang modeled EPROF [12] which is a fine energy profiler for smartphone applications directly helps an application developer in the energy optimization application cycle. Based on the power model FSM [14] EPROF has the ability to analyze the state of the application for asynchronous energy modeling energy characteristics, the state of the tail components with a granularity level routine.

The main objective of this solution is to determine where is the energy Spent inside applications, The precise identification of the responsible consumption of such entity to such a moment is hard to get, this is due to the asynchronous behavior of the energy (instantaneous power drawn from a component that cannot be linked to the current) [12].

Several components, such as disk, WiFi, 3G, GPS, in smartphones, exhibit the behavior of the tail of power [15-17], where activities into a single entity, (a routine for example), can trigger a component into a state of high power and remain in this state energy beyond the end of the routine.

The pivot point of this approach is the evaluation of the energy consumed by the input / output (I / O), and tail energy. I / O and tail are considered as major energy consumers. Such behavior is asynchronous. Several elements (GPS, Wi-Fi, SD Card, and 3G) have a state for tail energy [15, 16].

The operating systems for smartphones can switch to an inactive state after a period of rest, but applications must stay up [18].

The Wakelocks (processes that can force the CPU to stay active.) is another example of the power of smartphones into asynchronous appearance. The energy consumed (because of the devices wake up) is particularly important. It can help to track down bugs Wakelocks [19] (e.g. Facebook bugs [20], E-Mail bug, Android [21, 22], and Location listening bugs [23]).

The program can be optimized by restructuring the code in the process at the thread and under programs for remedial the problem of energy that can be the separation of the energy of the rest of the tail, the developer must understand the electrical behavior to well manage energy.

Table 1 provides a summary of the energetic consumption of five popular applications [12]:

Applications	Execution time	Percentage of battery consumption
Browser	30s	35 %
Angrybirds	28s	37 %
Fchess	33s	60 %
Nytimes	41s	75 %
Mapquest	29s	60 %

Table 1: energetic consumption of popular applications[12]

The energy consumed by the IOs can be optimized using the packages.

Note: The Input-Output consume a lot of energy.

The experiments showed the key observations on the energy consumed in the most popular applications:

- The majority of energy is consumed by the graphical display.
- The majority of the energy is dissipated by a few Input / Output.
- The CPU consumes very little power even when running thousands of routines.
- Minimizing the number of packets minimizes energy consumption.
- Table 2 provides a summary of some applications on the energy consumed towards the IOs [12]:

Applications	packages	IO used / total input-output
Handset:tytn2 runningWM6.5		
pslide	3 (3 Disk)	2/21
pup	3 (3 NET)	3/32
Handset:magic running Android		
syncdroid	4 (1 NET, 3 DISK)	8/0.9K
streamer	3 (3 NET)	4/1.1K
Handset:passion running Android		
browser	3 (2 Net, 1 GPS)	5/3.4K
angrybirds	4 (3 NET, 1 GPS)	5/2.2K
fchess	2 (2 NET)	7/3.7K
nytimes	2 (1 NET, 1 GPS)	16/6.8K
mapquest	3(2NET,1GPS)	14/7.1K
pup	1 (1 NET)	3/1.1K

Table 2 Energy breakdown summary per application [12]

Performance profiling is a long studied topic. The profiling run time has been given to the application level [24, 25, 26] to keep track of the call graph and estimate execution time routines, object-oriented languages [27, 28] and in the kernel [29].

Eprof is a model based on the use of power; it does not reflect the behavior of asynchronous power found in modern smartphones.

D. RAPL

In 2012, Marcus Hahnel, Bjorn Dobel, Marcus Volp and Hermann Hartig, developed the RAP L (Running Average Power Limit). RAPL is an approach which allows a comparison of the two mobile applications and sectors shows that they show different energy consumption while offering similar services.

The paper [30] introduced the notion of RAPL because that manual instrumentation is coarse and is at the expense of the power of the device, it remains inexact to calculate the energy consumed by tail applications. The authors of the article [31] tries to find approaches to energy efficiency create templates that adapt the behavior of applications, to measure the energy required to decode an image and choosing the path for the result a query.

The contribution of the paper is choice of path for the query result to calculate the energy consumed by the device drivers because an un-optimized driver will use more CPU resources, so there will be more energy consumption.

Hardware devices such as hard disks or network interfaces consume energy,

The proposed idea is to measure not only the energy consumption of the device, but also to measure the energy that the driver for this device consumes in terms of computing power.

RAPL is based on energy sensors available in the latest Intel processors to measure the energy consumption and to take account of energy consumption in the software components.

The use of RAPL infrastructure serves to characterize the energy costs and decode video tranches and choice of path for the query result.

Example of coding and decoding image: The encoder delivers two versions: a high quality (choice1) and another version of lesser quality that can be decoded with low consumption (choice2), the decoder will decide based on budget battery for choice between 1 or 2.

Choice of path for the query result: Generally the System Management database has several choices to achieve the result of a query.

The goal is to enable the DBMS to select not only the combination of the operator that calculates the result as quickly as possible, but also one that consumes less energy. [32]

The following limitations of RAPL model are:

- RAPL provided sensors that measure power consumption at the CPU and memory, but the problem resides in the technics and the impossibility of calculating consumption of IOs devices.
- RAPL must have precise information on the video split for minimizing energy costs which render it difficult to put into practice.

E. Appscope

Appscope [34] is a system that automatically evaluates energy applications running on Android smartphones consumption. Its design is based on monitoring the Android kernel at a microscopic level.

The objective of AppScope model is to measure the energy application system for applications that uses power equipment models and usage statistics for each hardware component.

The AppScope tool accurately estimates the energy consumption of Android applications. The system analyzes the traces of applications and system AppScope collects based on event-based approach [34] use information. AppScope estimated accurately and in real time, the use of hardware at a microscopic level.

AppScope was developed with the Linux 2.6.35.7 kernel. SystemTap version 1.3 [35] also uses K probes and data collection for the evaluation. All evaluations are conducted on HTC Google Nexus One (N1; Qualcomm QSD 8250 Snapdragon 1GHz, Super LCD 3.7 inch) [36] with the Android platform version 2.3

AppScope can be used on an Android-based software system without changing the device's limitation of the online method may possibly be overcome by using an online approach that uses a control unit battery (BMU) [37, 38] which is integrated in smartphones.

AppScope allows the detection of events that are pertinent to the operation of a hardware component such as CPU frequency change, Packages transmission and input / output (IOs)

Referring to K probes (probes K [39] which is used to monitor the behavior of system calls) AppScope is compiled as a kernel module and dynamically controlled.

Table 3 explains how AppScope interact with the various components [34]:

Component	Methods used par Appscope
CPU :	AppScope detects the switching process by tracking a wake event by sched_switch ()
Wi-Fi	The energy consumption of the LAN varies depending on the flow of packets (e.g. packets transmitted per second)
3G	The interface énergie3G consumption depends on the state of the CRR (CRR: The Radio Resource Control) [34]
LCD	The energy consumption of an LCD screen is proportional to the display brightness and display time.
GPS	AppScope monitors "LocationManager calls" and calculate the duration of activation of GPS

Table 3 equipments usage[34]

Notes:

When GPS is activated, AppScope account requests location for Location Manager. The count is then used to estimate the energy consumption for each process application.

For video experiences, instrumentation is done by Roitzsch adapted in [33].

Limitations:

- AppScope generates less accurate when the presence of the tail energy of the influence on the accuracy of the results generated by AppScope results.
- The precision of AppScope depend of the electrical aspect of the's unit in question.
- AppScope is operational in a limited number of processor architectures and do not take into account the multiple core architecture.

3. Analysis and comparison

The authors of the Tailender model had studied and tried to minimize the energy consumption, more particularly of 3G, GSM and Wi-Fi for applications that can tolerate a slight delay. The authors believed that the communication is the main cause of energy consumption. The Cinder model was set up to allow users to manage the limited resources of the device. It permits to model the energy consumption and its partitioning. The energy management is done using the principle of reserves and taps.

The difference between Cinder and tailender lies in the sources of energy consumption. The Tailender manages energy consumption equipment three mentioned above (3G, GSM, Wi-fi) with respect to applications that can tolerate a slight delay. Cinder is a system that manages consumption energy any source of consumption can be optimized. The Cinder model prevents the persistence of malicious applications. Finaly it maximizes the active state of the Smartphone for emergencies.

In 2012, Article [6] modeled Eprof which allowed having a clear identification of where energy is spent within the applications. It determines the responsible consumption of such entity such a moment.

- Goals attended by Tailender and Eprof are complementary. They focus on the energy consumed by the IOs, and also energy of the tail. Unlike Cinder model Eprof enter inside applications, On the contrary, Eprof cannot handle the security side. such as the detection of the use of malicious applications and management of the emergency call
- The operating limit of average power (RAPL) was defined in 2012 and this method allows comparison of mobile applications and proves that the two sectors show different energy consumption while offering similar services.
- The RAPL model minimizes the energy of the tail. It offer several options to reduce the energy consumed, its operation is complementary to Eprof model. it identifies where the energy is spent. The RAPL method has not addressed the security.
- In June 2012, AppScope was put into practice to automatically evaluate energy applications running on Android Smartphone consumption. The authors of the article have shown the effectiveness of the proposed method. They have determines that the IOs are the main sources that influence the quality of AppScope's Service.

In this presentation we have identified two groups. The group "manager" manages resources such as Tailender, Cinder and RAPL. The group "Feedback" that have feedback and information on the precise identification of energy path such as: Eprof and AppScope.

The following table includes the relevant criteria with respect to the different models studied in our investigation:

- Quality of the energy model: This criterion enlightens the effort of the work, spent in providing an effective energy model.
- Hardware performance optimization: This criterion shows the quality of the energy model regarding the hardware consumption of energy.
- Software performance optimization: This criterion shows the quality of the energy model regarding the software implementation.
- Impact on quality of Service: This criterion details the damage achieved by the method on the QoS in order to reduce energy consumption.
- Security: The security criterion shows the ability of the method to prevent malware to reduce smartphone lifetime.
- Developer assistance: This last criterion is used to enlighten the effort of the authors to help the developer in order to reduce the energy consumption of this software.

We note that the models of the first type (tailender, Cinder and RAPL) achieved relevant results for the following three criteria: performance optimizations hard, impact on quality of service and security.

However, models of the second type (Eprof and AppScope) have shown their effectiveness in the other three criteria: quality of the energy model, performance optimization applications and helps the developer.

The Cinder model is the only one who has made three times the best result for criteria 2, 4 and 5 (see table).

The AppScope model show high efficiency for the optimizations performance test of applications. AppScope is more recent but has forsaken the track as the security; this is due to the fact that it is not its goal.

	Quality of the energy model	Performance optimisations hard	Performance optimizations Application	Impact on Quality of service	security	Help Developer
TailEnder	+	++	NA	-	NA	NA
Cinder	+	++	NA	--	+++	NA
Eprof	++	+	+	-	NA	++
RAPL	+	++	NA	-	+	+
AppScope	+	NA	+++	-	NA	+

Table 4 : Aspects of models overlooked the criteria of quality and performance

QOS and security tend to be considered less over time, while optimizing application is a problem that seems to be leading

4. Conclusion and perspective

In this paper, we studied the optimization problem of theenergy consumed into a smartphone. First of all, we presented the assessment and measurement of the energy cost, then the problems encountered in the methods used for energy optimization. We delt with five major models: Tailender, Cinder, Eprof, RAPL, and AppScope.

We tried to classify them using six different Criteria. Several parameters are taken into account; the importance of criteria is related to levels of consumption as well as evaluating the energy cost. The results presented allow us knowing the main sources responsible of the energy consumption.

The models that have appeared after EPROF wanted to be more specific energy consumption, we note that the latest models do note take into account the energy of the tail so that it influence permanently on the activities of software and models.

In future work, we will continue to evaluate thesis models trying to hybridize to the extent possible in order to benefit from the advantages of each.

We are interested later by estimating the energy consumed in a mobile device and that is why the result of our research will focus on AppScope and Eprof.

REFERENCES

[1] G.P. Perrucci, « Energy Saving Strategies on Mobile Devices », *Multimedia Information and Signal Processing - Aalborg University*, Aalborg, janvier 2009, 1 - 150

[2] N. Balasubramanian and et.al., “Energy consumption in mobile phones: a measurement study and implications for network applica-tions,” in Proc of IMC, 2009

[3] C.-C. Lee, J.-H. Yeh, and J.-C. Chen. Impact of inactivity timer on energy consumption in wcdma and cdma2000. In Proceedings of the Third Annual Wireless Telecommunication Symposium (WTS). IEEE, 2004.

[4] J.-H. Yeh, J.-C. Chen, and C.-C. Lee. Comparative analysis of energy-saving techniques in 3gpp and 3gp2 systems. In Transactions on Vehicular Technology, volume 58, pages 432–448. IEEE, 2009

[5] X. Chuah, M.;Wei Luo; Zhang. Impacts of inactivity timer values on umts system capacity. In Wireless Communications and Networ-king Conference (2002), volume 2, pages 897–903. IEEE, 2002

[6] A. Roy, S. M. Rumble, R. Stutsman, P. Levis, D. Mazieres, and N. Zeldovich, “Energy management in mobile devices with the Cinder operating system,” in Proc. of EuroSys, 2011.

[7] H. Zeng, C. S. Ellis, A. R. Lebeck, and A. Vahdat. Ecosystem: managing energy as a first class operating system resource. In AS-PLOS-X: Proceedings of the 10th international conference on Archi-tectural support for programming languages and operating systems, pages 123–132, New York, NY, USA, 2002. ACM.

[8] H. Zeng, C. S. Ellis, A. R. Lebeck, and A. Vahdat. Currentcy: A unifying abstraction for expressing energy management policies. In In Proceedings of the USENIX Annual Technical Conference, pages 43–56, 2003.

[9] R. Fonseca, P. Dutta, P. Levis, and I. Stoica. Quanto: Tracking energy in networked embedded systems. In R. Draves and R. van Renesse, editors, OSDI, pages 323–338. USENIX Association, 2008.

[10] J. Flinn and M. Satyanarayanan. Powerscope: A tool for profi-ling the energy usage of mobile applications. In WMCSA ’99: Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications, page 2,Washington, DC, USA, 1999. IEEE Computer Society.

[11] H. Zeng, C. S. Ellis, A. R. Lebeck, and A. Vahdat. Ecosystem: managing energy as a first class operating system resource. In AS-PLOS-X: Proceedings of the 10th international conference on Archi-tectural support for programming languages and operating systems, pages 123–132, New York, NY, USA, 2002. ACM

[12] Pathak, Abhinav,Hu, Y. Charlie,Zhang, Ming.Proceedings of the 7th ACM european conference on Computer Systems - EuroSys '12 : Where is the energy spent inside my app?Fine Grained Energy Accounting on Smartphones with Eprof

[13] “Mobile app internet recasts the software and services land-scape.” URL: http://tinyurl.com/5s3hhx6

[14] “Apples app store downloads top 10 billion.” URL: http://www.apple.com/pr/library/2011/01/22appstore.html

[15] A. Pathak, Y. C. Hu, M. Zhang, P. Bahl, and Y.-M. Wang, “Fine-grained powermodeling for smartphones using system- call tracing,” in Proc. of EuroSys, 2011.

[16] N. Balasubramanian and et.al., “Energy consumption in mo- bile phones: a measurement study and implications for net- work applications,” in Proc of IMC, 2009

[17] L. Zhang and et.al., [1] http://fr.wikipedia.org/wiki/Consommation_%C3%A9nerg%C3%A9tique_d%27un_smartphone

[18]“Android powermanager:Wakelocks.” URL: http://developer.android.com/reference/android/os/PowerManager.html

[19] A. Pathak, Y. C. Hu, and M. Zhang, “Bootstrapping energy debugging for smartphones: A first look at energy bugs in mobile devices,” in Proc. of Hotnets, 2011

[20] “Facebook 1.3 not releasing partial wake lock.” URL: http://geekfor.me/news/facebook-1-3-wakelock/

[21] “Email 2.3 app keeps awake when no data connection is avai-lable.” URL: http://www.google.com/support/forum/p/ Google+Mobile/thread?tid=53bfe134321358e8

[22] “Email application partial wake lock.” URL: http://code.google.com/p/android/issues/detail?id=9307

[23] “Using a locationlistener is generally unsafe for leaving a permanent partial wake lock.” URL: http://code.google.com/p/android/issues/detail?id=4333

[24] S. L. Graham, P. B. Kessler, and M. K. McKusick, “gprof: A call graph execution profiler,” in Proc. of PLDI, 1982.

[25] G. C.Murphy, D. Notkin,W. G. Griswold, and E. S. Lan, “An empirical study of static call graph extractors,” ACM Trans. Softw. Eng. Methodol., vol. 7, April 1998

[26] J. Spivey, “Fast, accurate call graph profiling,” Software: Prac-tice and Experience, 2004

[27] M. Dmitriev, “Profiling Java applications using code hotswap- ping and dynamic call graph revelation,” in Proceedings of the 4th InternationalWorkshop on Software and Performance. ACM, 2004, pp. 139–150

[28] D. Grove, G. DeFouw, J. Dean, and C. Chambers, “Call graph construction in object-oriented languages,” ACM SIGPLAN Notices, vol. 32, no. 10, pp. 108–124, 1997

[29] “Oprofile.” URL: http://oprofile.sourceforge.net/news

[30] Universit, Technische, Dresden, Dresden « Measuring Energy Consumption for Short Code Paths Using RAPL », Vol 40 ,pp 13-17 , 2012

[31] D. C. Snowdon, S. M. Petters, and G. Heiser. Accurate on-line prediction of processor and memory energy usage under voltage scaling. In

Proceedings of the 7th International Conference on Embedded Software, pages 84–93, Salzburg, Austria, Oct 2007.

[32] Advanced Micro Devices. BIOS and Kernel Developer's Guide (BKDG) for AMD Family 15h Models 00h-0Fh Processors. 2012.

[33] M. Roitzsch. Slice-balancing H.264 video encoding for improved scalability of multicore decoding. In Proceedings of the 7th International Conference on Embedded Sofware, Salzburg, Austria, EMSOFT'07. ACM, 2007.

[34] Yoon, Chanmin,Kim, Dongwon,Jung, Wonwoo,Kang, Chulkoo, Cha, Hojung : AppScope: Application Energy Metering Framework for Android Smartphones using Kernel Activity Monitoring

[35] SystemTap, http://sourceware.org/systemtap.

[36] HTC Google Nexus One, http://en.wikipedia.org/wiki/Nexus_One

[37] M. Dong and L. Zhong. Self-constructive High-rate Sys- tem Energy Modeling for Battery-powered Mobile Sys- tems. In MobiSys, 2011

[38] W. Jung, C. Kang, C. Yoon, D. Kim, and H. Cha. Non- intrusive and online power analysis for smartphone hardware components. Technical Report, MOBED-TR-2012-1, Yonsei University, 2012.

[39] Kprobes, http://www.kernel.org/doc/Documentation/kprobes.txt