Ronny Hans, Daniel Burgstahler, Alexander Mueller, Manuel Zahn, and Dominik Stingl:
Knowledge for a Longer Life: Development Impetus for Energy-efficient Smartphone Applications. In:
Proceedings of the 4th International Conference on Mobile Services (MS 2015), Institute of Electrical and Electronics Engineers (IEEE), June 2015.

Knowledge for a Longer Life: Development Impetus for Energy-efficient Smartphone Applications

Ronny Hans, Daniel Burgstahler, Alexander Mueller, Manuel Zahn, and Dominik Stingl
Multimedia Communications Lab (KOM)
TU Darmstadt, Rundeturmstr. 10, 64283 Darmstadt, Germany
Email: Ronny.Hans@KOM.tu-darmstadt.de

***Abstract*—In recent years, there has been a rapid growth in the spread of smartphones and thus in the utilization of mobile applications. Such applications require a substantial portion of the available energy. Since a short battery lifetime has a very negative impact for the user experience, application developers should have the skills and the knowledge to avoid energy-inefficient applications. In this paper, we present a comprehensive survey of approaches and methods to reduce the energy consumption and thus help software developers to improve their applications.**

***Index Terms*—energy consumption, application, smartphone**

I. Introduction

Smartphones have become a constant companion and indispensable help in our daily lives. Thanks to a plethora of diverse applications (abbreviated as apps), they store our tickets when we travel, help to find our way, entertain us with music or videos, and keep us in touch with our colleagues and friends. Due to this important role of smartphones, nothing is more annoying than a shutdown due to an empty battery. Despite all technical development, energy remains a scarce resource and the most limiting factor in mobile devices. A recent study indicates that there is most likely no change of this fact in the near future [1].

A large share of energy consumption is caused by applications. Depending on the functionality of an app, they constantly present information on the display, download data, or use build-in sensors, such as sensor for the Global Positioning System (GPS) to determine their current position. Consequently, each of the used components consumes a substantial amount of the available energy. Among these components, the display, the CPU, the network interfaces and the GPS sensor have been identified as main consumers of energy. As a result, much effort has been made to investigate and identify the potential for energy savings of these components on a hardware level.

The focus of our work is primarily on application development. Thus we explicitly exclude a whole bunch of work, which is definitely interesting for energy improvements, but which is out of scope for application developers, such as modification of Hardware, optimization in *Medium Access Control*, aggregation of frames and packages, or the development of new protocols. Further we only focused on literature that provides enhancement potentials and proof their proposals with a quantitative evaluation.

Thus, in this paper, we give an overview of approaches and current developments to reduce the energy consumption of mobile applications. Thereby, we focus on useful knowledge, specifically for applications developers. In Table I we give an overview of the used operating systems in the respective related literature.

The remainder of the paper is structured as follows: First, in Section II we have a closer look on the major input/output device, the display. Thereby the focus lies on energy-specific improvement within graphical user interface (GUI). Later on, in Section III we present selected papers regarding energy improvements for wireless data transfer. In Section IV we present concepts to avoid the high energy consumption of the GPS sensor. Subsequently, in Section V we present concepts to improve the source code of mobile application. In Section VI we present offloading approaches and their advantage to relieve the local CPU of computation tasks. Afterwards, in Section VII, we give an overview of related work. Finally, we conclude our work with a summary in Section VIII.

II. Graphical User Interface

The central input and output device of a smartphone, where nearly all the user interactions take place, is the display. Thereby, the energy consumption of the display is mainly determined by the brightness [2], [3]. For all current display types, i. e., Liquid Crystal Display (LCD), Organic Light Emitting Diode (OLED) and Active-Matrix Organic Light-Emitting Siode (AMOLED), several scientific studies find a linear correlation between energy consumption and changes in brightness [4], [5], [6]. Since a general reduction of brightness is not feasible, e. g., for outdoor activities during the day, several more specific approaches are proposed to reduce the energy consumption.

To begin with, Dong and Zhong analyzed the influence of the displayed content on the energy consumption for OLED-based displays. They figure out that the energy consumption highly depend on display content. For displaying different colors, a different amount of energy is required. Thus, the graphical user interface (GUI) designers have a huge influence on the energy consumption. The authors developed different energy models to estimate the power consumption of the content to be displayed. Further, different transformation methods are proposed, e. g., the usage of a dark background color and a lighter foreground color. Using the transformation methods

Table I: Overview of Used Operating Systems

Operating System	Reference
Android (unspecified)	[9] [10] [11] [12] [8] [5] [13]
Android v1.x	[14] [4] [15]
Android v2.x	[6] [16] [17] [18] [19] [20]
Android v4.x	[21] [22] [23] [7]
Symbian	[24] [25] [26]
Windows Mobile (unspecified)	[10]
Windows Mobile 5.x	[27] [2]
Windows Mobile 6.x	[16] [28]

the authors evaluate the influence on the user satisfaction. With only small reduction of the user satisfaction, the energy consumption can be reduced by about 75% [2].

A similar idea to reduce the energy consumption of OLED-based displays is pursued by Li et al. Specificity in web applications, a huge amount of energy is required to display large light colored backgrounds. In contrast to Dong and Zhong the authors propose to change the source code of such applications that is stored at the web server. Therefore, a tool named Nyx is developed that is able to perform automatic transformations of color schemes of web applications. With such modified applications energy savings of up to 40% are possible, by only a minimal downgrade in user ratings of the web site . Nevertheless, 97% of the users are willing to accept the color transformation if they are on critical battery level [7].

Wee and Balan as well focus on the energy optimization of OLED displays. Thereby, they analyzed the brightness reduction of non-interesting regions of the display. The authors determine the area of interest in the center of the screen and dim the display towards the edge. The solution was evaluated with the mobile shooting game *Kwaak 3* by measuring the number of kills. Since there was no statistical significant difference, the authors consider no impact to the playability of the game. With this approach up to 11% energy reduction of the display and up to 4% total reduction of energy consumption can be achieved [8].

Regarding modern OLED-displays the selected papers show two main research directions. On the one hand brightness reduction in less interesting areas of the display and changes of the displayed color, e. g., to avoid light color areas which consume a high amount of energy.

III. Data Transfer

Energy efficiency for sending and receiving data is a research area that receives high attention. Nevertheless, a lot of approaches are useful for example for future protocol design or related topics, but they do not have a large impact for current application development. In the work at hand we focus on the selection of wireless interfaces, on scheduling of data transmission and on data compression.

Smartphones are equipped with several wireless interfaces. All with different technical characteristics and energy profiles. From an energy point of view it is inefficient if multiple interfaces are powered on while not needed, e. g., the energy consumption increase by 18.3% when both UTMS and Wi-Fi are powered on, compared with just UMTS being powered on [14]. Therefore, the decision for a specific interface plays an imported role.

Rahmati and Zhong focus on saving energy by choosing the appropriate wireless interface based on several contextual factors. The authors figure out that cellular and Wi-Fi networks have complementary energy profiles. The challenge for an efficient usage of battery resources is the decision in which point in time the Wi-Fi should be activated. Therefore, the authors formulate a statistical decision problem, which based on different context information such as time, history and cellular network conditions. With their approach the authors improve the average battery lifetime up to 39% [27].

In order to reduce the energy consumption of smartphones, Taleb et al. propose a dynamic switching between 3G and Wi-Fi. The authors aim to switch efficiently from a primary cellular network to an alternative Wi-Fi connection. Therefore, they developed energy models regarding to the size of the download and regarding to the effective download bit rate. Using this information an application was developed to switch dynamically between 3G and Wi-Fi. Compared to Wi-Fi only and 3G only usage, energy savings of 18% or 30%, respectively are possible [18].

Since cellular connections require more energy as Wi-Fi connections, it is beneficial to use Wi-Fi for downloads containing a large amount of data. In practical cases, where Wi-Fi is not ubiquitously available, scheduling of data transfer could be appropriate.

Especially for video streaming, prefetching seems to be a promising method to save energy. Therefore, Gautam et al. analyzed the potential energy savings of the Android prefetching app *Incoming*. This application enables video stream prefetching based on past video viewing behavior. With the proposed approach, a Wi-Fi connection is used to download videos in advance to avoid streaming via 2G or 3G. With a chosen overhead of unused content between 37% and 70%, energy savings up to 84% for prefetching over Wi-Fi compared to 3G are possible. For streaming over 2G up to 98% reduction of energy consumption is possible [20].

Besides the chosen wireless interface, also the signal strength influences the energy consumption. To save energy in 3G networks, Schulman et al. propose a scheduling algorithm that makes use of a high signal strength. Therefore, the authors analyze the energy consumption dependent on the signal strength. As a result, they find six times higher energy consumption per transferred bit when the signal strength is weak. To take advantage of a high signal strength, the authors developed a scheduling algorithm for two kinds of applications. On the one hand for sync applications, such mail or news and on the other hand for streaming applications. Therefore, the algorithm uses previous signal measurements. For sync applications flexible synchronization intervals are used. For streaming applications the algorithm modulates the traffic stream to match with the radio energy characteristic. Thereby, the proposed algorithm takes energy for communica-

tion as well as tail energy into account. With their simulation, the authors show energy savings of up to 10% for email synchronization and up to 60% for on-demand streaming [29].

Balasubramanian et al. as well use flexible synchronization intervals and perfecting. However, the authors suggest an approach to reduce the energy that is wasted in high-power states after the completion of a data transfer. With a measurement study the authors figured out that a large fraction of energy is wasted that way, e. g., 60% for 3G. The authors propose to save energy in two different ways. On the one hand with a delayed data transfer for delay-tolerant applications such as e-mail applications and news feeds. On the other hand the authors consider applications that benefit from prefetching, i. e., web search. With their simulations they show an energy reduction of up to 35% for e-mail applications, of up to 52% for news feeds and of up to 40% for web search [28].

To reduce unnecessary background traffic that causes a noticeable extra energy consumption, Burgstahler et al. analyze the energy impact of push and pull client notifications [30]. Depending on the acceptable latency on client notifications a pull based approach can help to save energy compared to a push based approach. In [31] the authors show an energy saving potential of up to 7% by the use of a transition approach. Depending on the users context the authors suggest to switch between these two notification paradigms to save energy and at the same time to ensure low notification delays.

To reduce the time a wireless interface is active, the reduction of the amount of transferred data could be a feasible solution. Therefore, Hans et al. analyze the potential energy savings by compressed data transmission while invoking web services. For their investigation they use UMTS. On the one hand, compression reduces the number of data packages and those the up-time of the wireless interface. On the other hand, the decompression requires more computational power and thus additional energy. The evaluation shows for small payload and recurring web service invocations higher energy consumption with compression because of the tail energy. For single web service invocations with payload sizes larger than 50 KB, energy savings with compressions are substantial. The authors show possible energy savings up to 21.5% [25].

To summarize, there are several possibilities to reduce energy consumption during data transmission. First of all, Wi-Fi should be used for large downloads or uploads. If possible, areas with high signal strength should be preferred for wireless cellular communication. Further, for delay-tolerant synchronization applications and video streaming, appropriate scheduling algorithms should be used. As well, data compression can be used to save energy. However, for frequent transmission of small amounts of data, application developers should be aware of the energy waste because of high-power states of wireless interfaces.

IV. Localization Methods

Within this section we analyze energy saving techniques with regard to localization methods. Since localization by the use of GPS is one of the most battery draining tasks on a smartphone, we see in these optimizations a big potential for energy conservation. Basically two approaches are feasible to reduce the energy consumption on localization tasks. First we have a closer look into substitution techniques that try to avoid the use of GPS for localization. The second approach is to optimize the usage of the GPS sensor.

A. GPS Substitution

Abdesslem et al. have introduced a sensing system named *SenseLess* [24]. The concept is based on reducing the use of sensors that have a high energy consumption and instead to increase the use of energy-efficient sensors. They also highlight the difference in energy consumption of several smartphone sensors, e.g., the energy consumption of the acceleration sensor is only about 15% of the GPS location sensor. They use the accelerometer to decide if a user is moving and thus a new location has to be sensed. During sensing the location they switch between GPS and Wi-Fi based localization. With their approach the authors were able to double the battery lifetime within the selected application scenario. Thereby, the accuracy won't be negatively affected.

For indoor localization Shafer and Chang follow a similar approach [12]. They use as well the accelerometer to determine whether a user is moving or not. The location is only determined directly after the user has stopped the movement. Directly afterwards a Wi-Fi scan is executed and RSSI (Received Signal Strength Indicator) fingerprints of the available Wi-Fi networks are taken. The result is sent to remote server that responds with the respective identifier of the physical location. With this approach the authors were able to reduce the power consumption by 80% compared to other solutions with an decrease of accuracy by only 5 %.

Zhuang et al. present an adaptive location sensing framework that is implemented as middleware for Android to support location based applications [15]. They also substitute energy intensive GPS location determination by less energy consuming techniques, e.g., network based location sensing. If possible the framework suppresses localization if the user is not moving. For the detection the accelerometer is used. Furthermore they synchronize the location request from multiple applications to reduce the sensing efforts. Additionally they adjust the sensing parameters when the battery is getting low. By the use of this framework the authors were able to improve the battery life of up to 75%.

B. Localization Usage Optimization

Lee et al. focus on the energy optimization for location based background applications [32]. The used mechanism varies the localization rate to reduce the energy consumption of localization efforts. Basically they introduce two algorithms to optimize the detection of the user context and thus to determine the localization rate. The first algorithm is named *StandStill Detection (SSD)* and decides when to execute the localization. During localization the accelerometer is switched off and during sleep phases the localization sensor is switched off. The accelerometer is used to detect movement within the

two sleep phases that differ in the sampling rate. The second algorithm is based on the classification between indoor and outdoor. By determining the average number of GPS satellites and the maximum number of GPS satellites in parallel received within a time window the algorithm is able to decide if the device is indoor or outdoor. They also give recommendations for configuration thresholds based on evaluation results.

A related application scenario is focused by Bulut and Demirbas that present an alternative implementation for Androids proximity alert service [21]. The middleware uses the distance to the point of interest and a detection of the users transportation mode. For the transport mode they decide between driving, walking and idle. Based on this knowledge the localization technique (GPS, Wi-FI or GSM) and the respective sensing interval are determined. An increasing distance to the point of interest allows an increase of the sampling rate. Further, a slower movement mode allows an additional increase of this value. Thus, it is possible to reduce the use of localization sample drastically, e.g., to reduce the GPS usage by up to 96%. With their implementation the authors were able to increase the battery live by more than 75% in the considered scenario.

Brouwers et al. propose an energy efficient Wi-Fi based localization approach [13]. A list of all available access points forms a fingerprint that allows a server to respond with the users location. Commonly this works very well in urban areas. In contrast to the traditional approach Brouwers et al. suggest to scan only a subset of the full Wi-Fi spectrum. The proposed solution stops scanning for further access points once a sufficient subset of access points is discovered. As further improvement a hysteresis margin approach is used to calculate similarities to a previous scan. This allows to stop scanning very fast in case of the user has not moved. The proposed solution shows energy saving potentials of up to 57%, depending of the used device.

V. Code Optimization

This section focuses on the optimization of source code created by the application developer. We focus our analysis on approaches that are feasible for developers of mobile applications using existing operation systems and platforms. Therefore, modification of the operation system itself is out of scope.

To identify dissipation of energy, Pathak et al. implement and evaluate a fine-grained energy profiler called *eprof*. With the use of this profiler, an approach to map the energy consumption back to program entities is presented. The authors analyze six popular smartphone applications and they figure out that 65% to 75% of the energy is spent on third-party advertisement. The results also show high energy consumption in I/O components such as 3G, Wi-Fi or GPS. Often, applications force such I/O components to stay in high power states for a long time. By restructuring the source code, energy savings between 20% and 65% are possible[16].

Also Zhang et al. aim to detect dissipation of energy within applications. Therefore, they developed a tool called *ADEL* (Automatic Detector of Energy Leaks) to find energy leaks within mobile applications. They focused on energy leaks which are caused by unnecessary network communication. According the authors, the sources for such a behavior are programming errors and wrong predictions for prefetching. Energy leaks are found in six out of 15 applications. By eliminating them a reduction of energy consumption by 52.7% to 62.1% is possible. The reduction is explained by avoiding useless downloads, poor download implementation, missing caching systems and aggressive prefetching [17].

Another source for dissipation of energy may be *Wakelocks*. Wakelocks are functions to keep smartphone components in an active state, e. g., they avoid the standby mode of the display while watching videos. However, unreleased wakelocks lead to unnecessary energy consumption since, for example, Wi-Fi or GPS stay active while they are unused. To save energy, Alam et al. analyze the optimal placement of wakelock functions within an application. The authors propose a data flow analysis to determine appropriate positions for the placement of *acquire* and *release* functions in the source code. Based on their analysis a reduction of energy consumption by 8% to 32% is possible [19].

Like already presented in Section III, delay-tolerant applications provide potentials to save energy. Xu et al. focus on optimizing of background email synchronization. Based on network and flash memory measurements, the authors claim that existing email clients handle synchronization tasks in an energy inefficient manner. To overcome energy inefficiencies, the authors recommend, for example, fast dormancy, decoupling of data transmission from data processing and reusing existing connections. Further, small caches for incoming mails help to process emails fast. By using these recommendations, it is possible to reduce the energy costs for email synchronization by 49,9% on existent email applications. Since other communication applications feature similar characteristics, the found improvements can be seen as general design principles for energy-efficient event handling [23].

In this Section we gave an overview for improvements for code optimization. Thereby, the results closely correspond to the previous sections. Especially the inefficient usage of network connections and the GPS sensor causes a lot of unnecessary energy consumption. By improving the source code of mobile applications, a large portion of energy could be saved.

VI. Cloud Services and Computation Offloading

Another promising approach for energy savings is the shift of computational tasks to remote servers. Thereby, different approaches are used. Currently very popular is to shift the whole computation into the cloud and just send the desktop content as a video stream to the user. Prominent examples are cloud gaming or Desktop-as-a-Service. On the other hand, specific computational intensive parts of applications are offloaded to remote servers which could be offered by cloud providers.

Properly the most popular frameworks for code offloading are proposed by Cuervo et al. [33] and Kosta et al. [34]. To automatically identify and offload certain functions of a mobile application, Cuervo et al. propose *MAUI*. Thereby, applications can be benefit from energy savings and performance improvements. Kosta et al. propose a framework to offload computation to the cloud named *ThinkAir* [34]. Thereby, the concept is based on smartphone visualization. Nevertheless, both works do not provide an extensive evaluation of energy improvements.

Kwon and Tilevich investigate in offloading for Android apps by shifting energy intensive functionalities to remote servers without partitioning the apps. Using common techniques for dividing the app in different parts, which are executed locally *or* on the remote side, may cause problems in case of network outages. The authors identify energy intensive functions and replicate them to the cloud. On runtime it is possible to switch between local and remote execution. This approach results in energy savings between 30% and 60% for four out of five third-party Android apps by keeping the original performance characteristics [11].

Hans et al. analyze the effects of cloud gaming to the energy consumptions of smartphones. The battery life time gains from shifting computational intensive tasks to cloud servers. Especially, for graphics intensive games the approach is promising. Nevertheless, the required constant video stream may have negative effects on battery lifetime. For a full controlled experimental environment, the authors developed their own server and client applications. They compared different game complexities and different video quality using local execution and remote executing. Using Wi-Fi for the remote connection, the authors find energy savings for cloud gaming between 12% and 38% [22].

Computation offloading is a promising approach, especially for computation intensive tasks. However, there are several challenges to keep in mind: First, the apps may require a permanently available network connection. Second, regarding the energy, there is trade-off between reducing the computational effort and increasing the network utilization. Finally, using remote execution, the network latency may affect the Quality of Service of the application [35].

VII. Related Work

Since smartphones entered the market of mobile devices, there has been a lot of scientific work to identify energy influencing factors as well as enhancement potentials for energy saving. In this section we present an overview of related surveys in this area.

Vallina-Rodriguez and Crowcroft [36] studied software solutions to reduce the energy consumption in mobile devices, e.g., energy aware operation systems, energy measurements and power models. The analyzed work ranges from 1999 to 2011. In contrast to our work the authors do not explicitly focus on improvement potentials for application developers. Since a lot of work in this area was published the last three years, the paper do not reflect current developments.

The technical report of Naik [37] outlines enhancement potentials for energy consumption in mobile devices by reflecting more than 200 papers. The study is structured from physical to application layer, enriched with many details. But this work is not targeted at the current smartphones generation and does not aggregate the information at the high level we do.

In the most recent survey from 2012 by Kennedy et al., firstly the major energy consuming components in mobile devices are identified by measurements [38]. Subsequently, they present enhancement potentials according to the identified components. The authors explicitly focused on energy savings within smartphone components and do not consider specific problems of application development. Thus, they do not take current development such as computation offloading into account.

The presented surveys cover a wide area of measurements, approaches and methodology for energy improvements within mobile devices. Nevertheless, with the best of our knowledge, there has been no survey which especially focuses on application development.

VIII. Conclusion

In the work at hand we presented a selection of scientific work, which address the topic of energy-aware development of mobile applications. For the graphical user interface, especially used conjointly with OLED-displays, large light colored areas should be avoided and replaced by other colors to save energy. In a variety of papers the energy consumption of the network interfaces plays an outstanding role for application development. In general, Wi-Fi should be used for large downloads. Further, delay-tolerant applications can profit by scheduling algorithms and data compression should be used. Using cellular networks, especially for small frequent communication, the so called tail energy should be taken into account. To save energy while using localization, the GPS sensor should be used wisely. The time when the sensor is active should be reduced or it should be replaced by other localization methods such as localization via Wi-Fi. Finally, current paradigms like cloud computing can help to reduce the energy consumption by offloading code or transferring complete services to remote servers. Especially with the spread of cloudlets, some interesting use cases for low-latency and computational intensive multimedia applications occur.

To summarize, there is no sole and general way to develop energy-aware mobile applications. All mentioned aspects should be analyzed in detail to reduce the energy consumption. The approaches presented in this paper offer great saving potentials, but require as well a thorough application development.

Acknowledgment

This work has been sponsored in part by the German Federal Ministry of Education and Research (BMBF) under grant no. 01IS12054, by E-Finance Lab e.V., Frankfurt a.M., Germany (www.efinancelab.de), and by the German Research Foundation (DFG) in the Collaborative Research Center (SFB) 1053

– MAKI. The authors are fully responsible for the content of this paper.

REFERENCES

[1] StrategyAnalytics, "Cellphone Energy Gap is Widening," https://www.strategyanalytics.com/default.aspx?mod=pressreleaseviewer&a0=4656, March 2015.

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

[3] G. P. Perrucci, F. H. P. Fitzek, and J. Widmer, "Survey on Energy Consumption Entities on the Smartphone Platform," in *Proceedings of the IEEE 73rd Vehicular Technology Conference*, 2011.

[4] A. Carroll and G. Heiser, "An Analysis of Power Consumption in a Smartphone," in *Proceedings of the USENIX Annual Technical Conference*, 2010.

[5] D. Kim, W. Jung, and H. Cha, "Runtime Power Estimation of Mobile AMOLED Displays," in *Proceedings of the Design, Automation & Test in Europe Conference & Exhibition*, 2013.

[6] R. Murmuria, J. Medsger, A. Stavrou, and J. M. Voas, "Mobile Application and Device Power Usage Measurements," in *Proceedings of the IEEE 6th Int. Conference on Software Security and Reliability*, 2012.

[7] D. Li, A. H. Tran, and W. G. Halfond, "Making Web Applications More Energy Efficient for OLED Smartphones," in *Proceedings of the 36th International Conference on Software Engineering*, 2014.

[8] T. K. Wee and R. K. Balan, "Adaptive Display Power Management for OLED Displays," in *Proceedings of the 1st ACM International Workshop on Mobile Gaming*, 2012.

[9] M. Altamimi, R. Palit, K. Naik, and A. Nayak, "Energy-as-a-Service (EaaS): On the Efficacy of Multimedia Cloud Computing to Save Smartphone Energy," in *Proceedings of the IEEE 5th International Conference on Cloud Computing*, 2012.

[10] H. Falaki, D. Lymberopoulos, R. Mahajan, S. Kandula, and D. Estrin, "A First Look at Traffic on Smartphones," in *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, 2010.

[11] Y.-W. Kwon and E. Tilevich, "Energy-Efficient and Fault-Tolerant Distributed Mobile Execution," in *Proceedings of the IEEE 32nd International Conference on Distributed Computing Systems*, 2012.

[12] I. Shafer and M. L. Chang, "Movement Detection for Power-Efficient Smartphone WLAN Localization," in *Proceedings of the 13th ACM International Conference on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*, 2010.

[13] N. Brouwers, M. Zuniga, and K. Langendoen, "Incremental Wi-Fi Scanning for Energy-Efficient Localization," in *2014 IEEE International Conference on Pervasive Computing and Communications*, 2014.

[14] H. Petander, "Energy-Aware Network Selection Using Traffic Estimation," in *Proceedings of the 1st ACM Workshop on Mobile Internet through Cellular Networks*, 2009.

[15] Z. Zhuang, K.-H. Kim, and J. P. Singh, "Improving Energy Efficiency of Location Sensing on Smartphones," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, 2010.

[16] A. Pathak, Y. C. Hu, and M. Zhang, "Fine Grained Energy Accounting on Smartphones with Eprof," in *Proceedings of the 7th ACM European Conference on Computer Systems*, 2012.

[17] L. Zhang, M. S. Gordon, R. P. Dick, Z. M. Mao, P. Dinda, and L. Yang, "ADEL: An Automatic Detector of Energy Leaks for Smartphone Applications," in *Proceedings of the 8th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, 2012.

[18] S. Taleb, M. Dia, J. Farhat, Z. Dawy, and H. Hajj, "On the Design of Energy-Aware 3G/WiFi Heterogeneous Networks Under Realistic Conditions," in *Proceedings of the 27th International Conference on Advanced Information Networking and Applications Workshops*, 2013.

[19] F. Alam, P. R. Panda, N. Tripathi, N. Sharma, and S. Narayan, "Energy Optimization in Android Applications through Wakelock Placement," in *Design, Automation and Test in Europe Conference and Exhibition*. IEEE, 2014.

[20] N. Gautam, H. Petander, and J. Noel, "A Comparison of the Cost and Energy Efficiency of Prefetching and Streaming of Mobile Video," in *Proceedings of the 5th Workshop on Mobile Video*, 2013.

[21] M. F. Bulut and M. Demirbas, "Energy Efficient Proximity Alert on Android," in *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops*, 2013.

[22] R. Hans, U. Lampe, D. Burgstahler, M. Hellwig, and R. Steinmetz, "Where Did My Battery Go? Quantifying the Energy Consumption of Cloud Gaming," in *IEEE 3rd International Conference on Mobile Services*, 2014.

[23] F. Xu, Y. Liu, T. Moscibroda, R. Chandra, L. Jin, Y. Zhang, and Q. Li, "Optimizing Background Email Sync on Smartphones," in *Proceedings of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, 2013.

[24] F. B. Abdesslem, A. Phillips, and T. Henderson, "Less is More: Energy-Efficient Mobile Sensing with SenseLess," in *Proceedings of the 1st ACM Workshop on Networking, Systems, and Applications for Mobile Handhelds*, 2009.

[25] R. Hans, M. Zahn, U. Lampe, R. Steinmetz, and A. Papageorgiou, "Energy-efficient Web Service Invocation on Mobile Devices: The Influence of Compression and Parsing," in *Proceedings of the 2nd International Conference on Mobile Services*, 2013.

[26] J. K. Nurminen, "Parallel Connections and Their Effect on the Battery Consumption of a Mobile Phone," in *Proceedings of the 7th IEEE Conference on Consumer Communications and Networking Conference*, 2010.

[27] A. Rahmati and L. Zhong, "Context-for-Wireless: Context-Sensitive Energy-Efficient Wireless Data Transfer," in *Proceedings of the 5th International Conference on Mobile systems, Applications and Services*, 2007.

[28] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, "Energy Consumption in Mobile Phones: A Measurement Study and Implications for Network Applications," in *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*, 2009.

[29] A. Schulman, V. Navda, R. Ramjee, N. Spring, P. Deshpande, C. Grunewald, K. Jain, and V. N. Padmanabhan, "Bartendr: A Practical Approach to Energy-aware Cellular Data Scheduling," in *Proceedings of the 16th Annual International Conference on Mobile Computing and Networking*, 2010.

[30] D. Burgstahler, U. Lampe, N. Richerzhagen, and R. Steinmetz, "Push vs. Pull: An Energy Perspective," in *Proceedings of the 6th IEEE International Conference on Service Oriented Computing And Applications*, 2013.

[31] D. Burgstahler, N. Richerzhagen, F. Englert, R. Hans, and R. Steinmetz, "Switching Push and Pull: An Energy Efficient Notification Approach," in *Proceedings of the 3rd IEEE International Conference on Mobile Services*, 2014.

[32] C. Lee, G. Yoon, and D. Han, "A Context-based Energy Optimization Algorithm for Periodic Localization in Smartphones," in *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems*, 2012.

[33] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "MAUI: Making Smartphones Last Longer with Code Offload," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, 2010.

[34] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "ThinkAir: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *Proceedings of the 31st Annual IEEE International Conference on Computer Communications*, 2012.

[35] U. Lampe, Q. Wu, R. Hans, A. Miede, and R. Steinmetz, "To Frag Or To Be Fragged - An Empirical Assessment of Latency in Cloud Gaming," in *Proceedings of the 4th International Conference on Cloud Computing and Services Science*, 2013.

[36] N. Vallina-Rodriguez and J. Crowcroft, "Energy Management Techniques in Modern Mobile Handsets," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 179–198, 2013.

[37] K. Naik, "A Survey of Software Based Energy Saving Methodologies for Handheld Wireless Communication Devices," Dept. of ECE, University of Waterloo, Tech. Rep. No. 2010-13, 2010.

[38] M. Kennedy, A. Ksentini, Y. Hadjadj-Aoul, and G.-M. Muntean, "Adaptive Energy Optimization in Multimedia-Centric Wireless Devices: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, pp. 768–786, 2013.