Software-based Energy Requirement Measurement for Smartphones

Maximilian Schirmer, Hagen Höpfner

Mobile Media Group
Bauhaus-Universität Weimar
Bauhausstraße 11
99423 Weimar
maximilian.schirmer@uni-weimar.de
hoepfner@acm.org

Abstract: Energy is one of the most limiting factors for Smartphone-based computing. There exist various approaches for optimising or reducing, respectively, energy requirements in this context. In order to evaluate these techniques, one has to comparatively measure energy requirements. Most precise energy measurement is subject to attaching measurement hardware to the energy demanding system components. In this paper, we survey alternative software-based measurement approaches that utilise sensor data gathered from built-in energy-related sensors. We illustrate how they can be used and discuss their applicability while considering different operating systems and energy sensors.

1 Introduction and Motivation

According to a report published by Gartner [GP12] in February 2012, "smartphone sales soared in the fourth quater of 2011 with 47 percent growth". Since this growth is a continuous process (cf., [GP10]), smartphone computing becomes more and more important. However, according to the authors of [SH11], energy is the most limiting factor for the use of mobile devices. In order to prolong their uptime, a rethinking in many software development issues [HB11] is required. Various researchers try to overcome this limitation by optimising or reducing, respectively, energy requirements. Their approaches reach from hardware-based optimisations like sleep modes, performance scaling or the use of dedicated hardware [BMSS10], over resource substitution strategies [VOH+04, SH11] and energy-efficient algorithms, [KLGT09, BHRM11] to energy-aware software development [BH08]. There also exist various techniques for evaluating the mentioned approaches. The most precise one utilises measurement hardware connected to the internal circuitry. Its obvious disadvantage is that disassembling the devices is required. A less precise approach is the use of pre-calculated energy profiles. In this paper, we discuss a software-based measurement approache that utilises the energy information provided by the smartphones' operating system. This information is gathered from energy sensors built into the smartphone and can be used for monitoring software energy requirements while using the software.

The remainder of this paper is structured as follows: Section 2 discusses related work. Section 3 briefly explains the energy-related basic principles of physics. Section 4 presents the software-based techniques to measure energy. Section 5 illustrates the evaluation and discusses evaluation results. Finally, Section 6 summarises and concludes the paper and gives an outlook on future research.

2 Related Work

The research presented in this paper focusses on energy requirement ascertainment techniques. The authors of [HB10] introduced an experimental setup for measuring the energy requirements of the central processing unit (CPU) and the memory of an Atmel AVR micro controller-based system. Therefore, they connected measurement hardware to the hardware components of the system and recorded data about voltage drops at a sense resistor for the duration of a software execution. Energy is then calculated, following Ohm's and Kirchhoff's law, by calculating the integral of the curve defined by the data. For the experiments presented in [BH11], this setup was aligned to the measurement of smartphone energy requirements. The major disadvantage of this approach is twofold. Firstly, the measured device needs to be disassembled. Secondly, the measurement setup is quite heavy as it includes a digital oscilloscope and a device for recording measurement data.

Nokia provides a tool called Nokia Energy Profiler[1] and an application programming interface (API)[2] that enables developers to monitor power consumption, as well as battery voltage and current on Nokia S60 devices.The authors of [KLGT09] used this approach. In contrast to the Nokia tool, the authors of [ZTQ+10] evaluated their software-based energy measurement approach called PowerTutor [Pow11] using hardware-based reference measurements. The basic idea is to learn a cost model regarding the power consumption and energy demands for the hardware of a mobile device, and to use the learned model later on for judging on the energy demands of a software system. The benefit of this strategy is that it works without any modifications to the used devices. However, it is limited in two ways: On the one hand, it is less precise than the hardware based measurement. On the other hand, it is only applicable if the used devices are included in the learned model.

Our approaches presented in this paper relax those limitations. We directly measure real energy data during processing as it is done in the hardware-based approach, but do not need any hardware modifications. Furthermore, as discussed in [HSB12], the software-based measurement is almost as precise as the hardware-based approach.

[1] http://www.developer.nokia.com/Resources/Tools_and_downloads/Other/Nokia_Energy_Profiler/

[2] http://www.developer.nokia.com/Resources/Tools_and_downloads/Other/Nokia_Energy_Profiler/External_APIs.xhtml

3 Principles of physics

All electricity effects depend on moving charges and therefore on electrons and protons. The electric current (I), measured in ampere (A), represents the amount of electric charge that passes a point in an electric circuit per unit time. The electric voltage (U), measured in volt (V), is the potential difference between two points. The electric power ($P = U \cdot I$), measured in Watt (W), is the rate at which electric energy is transferred by an electric circuit. The energy (E), measured in Joule (J), is a function of voltage and ampere over time. In other words it is the consumed power for a given time.

Some smartphones include an electrical current sensor, which provides in combination with up-to-date voltage data from the device's battery all required data to calculate electrical power and energy using the following formula: $E = \int P(t)dt = \int U(t) \cdot I(t)dt$. The equation can be reduced to a simple summation when electrical current and voltage values are sampled equidistantly at frequency f: $E = \sum_{j=0}^{n} U(j) \cdot I(j) \cdot f^{-1}$. When sampled at 1 Hz, the equation is simplified to: $E = \sum_{j=0}^{n} U(j) \cdot I(j) \cdot 1\,\mathrm{s}$. There are, however, two major drawbacks to this approach: *resolution* and *availability*. In our experience, the data provided by the smartphones is updated only at a very low frequency. Values tend to remain stable for a few seconds, even when there is actually a high-frequency oscillation going on. Compared to the hardware-based approach, where sampling rates in the range of a few kHz can be achieved [BH12], it is possible to miss short-time fluctuations.

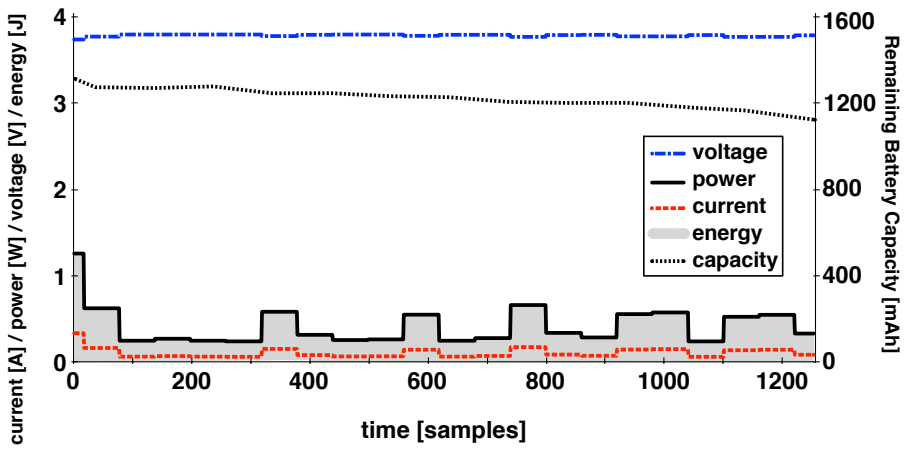


Figure 1: Data-based software measurement example.

Figure 1 illustrates the low *temporal resolution* with an exemplary measurement from a HTC Sensation. Samples were logged at a frequency of 1 Hz. The diagram shows the progression of electrical current, voltage, and derived from this data, electrical power. Furthermore, we included the electrical energy as the area below the electrical power curve, as well as the progression of remaining battery capacity. It is easy to see that peaks (in this case caused by GPS requests, in an evaluation that was part of [HS12]) can be detected, but changes are abrupt and it is hard to detect short-time fluctuations.

4 Software-based Measurement

Software-based measurement of power consumption and energy demands presents an interesting alternative to hardware-based approaches. Because no additional hardware components are needed, measurements in the field and with a short preparation phase can be conducted easily. Weight-intensive hardware components for a measurement environment are no longer required. In our previous experiments (e. g. [HSB12, BH11]), we used hardware oscilloscopes that typically weigh 5 kg, including a power supply. This of course hinders real-world evaluations of energy data on highly mobile devices.

In contrast to this, the software-based approach measures power and energy characteristics through the mobile device's internal power management software components. Normally, these software components are used internally to calculate the remaining uptime of the device. This information allows the device to initiate power saving methods when the remaining battery uptime falls below a given threshold.

However, most smartphone SDKs (we have experience with the iOS and Android platforms) only provide high-level access to interpreted power management values in the form of percentage of full charge, or remaining battery capacity in mAh. While this information is sufficient to react to emergency situations where the device is about to shut down because of an empty battery, it definitely does not suffice to make clear statements about the energy demands of software or software components.

Regarding the type of the power and energy data that is gathered with a software-based measurement, two approaches can be distinguished: (1) Measuring remaining battery capacity, and (2) Measuring battery voltage and discharge current.

4.1 Approach 1: Remaining Capacity Measurement

Information about the remaining battery capacity is crucial for using mobile devices. Normally, this information is also distributed to mobile applications through the SDK. In our experiments with the iOS platform [SH11], we were able to gather remaining capacity updates with a granularity of 1 mAh. On the Android platform, remaining capacity is only available as an interpreted percentage value on some devices. This of course results in a very low temporal resolution of the gathered energy data. On other Android devices, remaining capacity information is also available in the form of "real" mAh.

As shown in Figure 2, this varying availability results from the different architectural layers where energy data is gathered from. The simplest way is to query the remaining capacity on the API level. While this is sufficient to react on transitions between battery level categories (e. g. on a continuum between fully charged – empty), it is not sufficient to measure the energy requirements of mobile applications or hardware use because of the extremely poor temporal resolution.

On lower levels of the architecture, more granular and more frequently updated data is available. Through the IOKit framework, the iOS platform exposes the momentary and

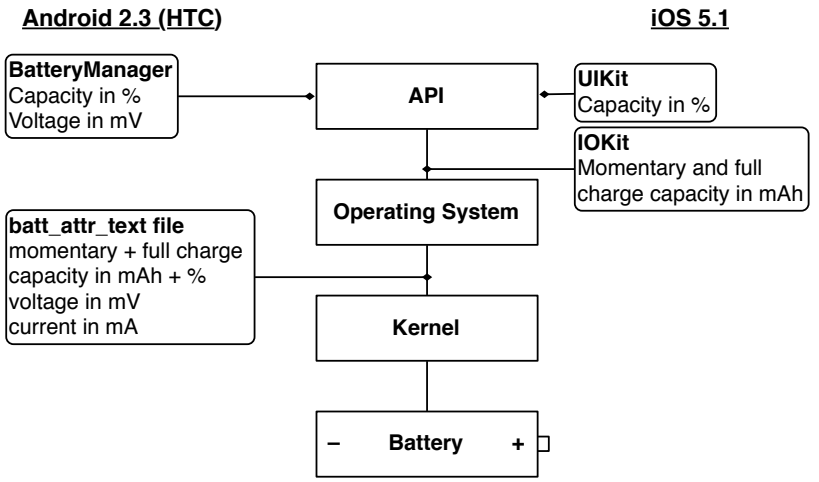


Figure 2: Availability of energy-related data on the Android 2.3 and iOS 5.1 platform.

full battery capacity in mAH. On the Android platform, this data is distributed through the file system and can be accessed by parsing the respective files. On most HTC devices, a single virtual file contains all relevant data and exists at the path `/sys/class/power_supply/battery/batt_attr_text`. The temporal resolution of this data is solid, in our experiments we have been able to achieve a frequency of about 1 Hz for value updates.

However, measuring energy data based on the remaining battery capacity is vulnerable to the fact that the discharge behaviour and health condition of the device's battery may be unknown. Damaged or old batteries tend to drop abruptly to a depleted state after they discharged normally to a certain capacity threshold. This would falsify gathered data gravely.

4.2 Approach 2: Voltage and Discharge Current Measurement

The second approach is only available on the Android system and involves a separate measurement of electrical current and voltage over a certain time span. Some devices expose the electrical discharge current that flows through the battery. This of course only provides valid data when the device is currently not charging. Otherwise, the current sensor reports the charging current, which is not directly related to the power consumption of the device. Some devices distinguish charge and discharge current by signing the value, others provide separate values, or just one of them. Measuring electric current relies on the availability of the respective sensor. While this sensor is common among smartphones, it is not guaranteed to be included in every device. And while some devices include the sensor, they do not expose its values.

Table 1 presents an overview of our observations regarding the availability of electrical current sensors among our collection of testing devices. This short excerpt shows that the broad real-world application of measurements based on electrical current sensors requires a careful selection of testing devices.

Model	**Electrical Current Sensor data available**
G1 (HTC Dream)	no
HTC Desire	yes
Google Nexus One	yes
Google Nexus S	no
HTC Sensation	yes

Table 1: Availability of electrical current sensors.

Despite this drawback of availability, the approach allows to compute energy demand directly from the gathered data, and does not involve the undocumented processing that results in remaining capacity information.

In a similar way as for the first approach, temporal resolution is an issue here, as well. On our experiments with the Android platform, both voltage and current regularly remain stable for intervals of up to $10\,\mathrm{s}$, resulting in an update frequency that can be as low as $0.1\,\mathrm{Hz}$.

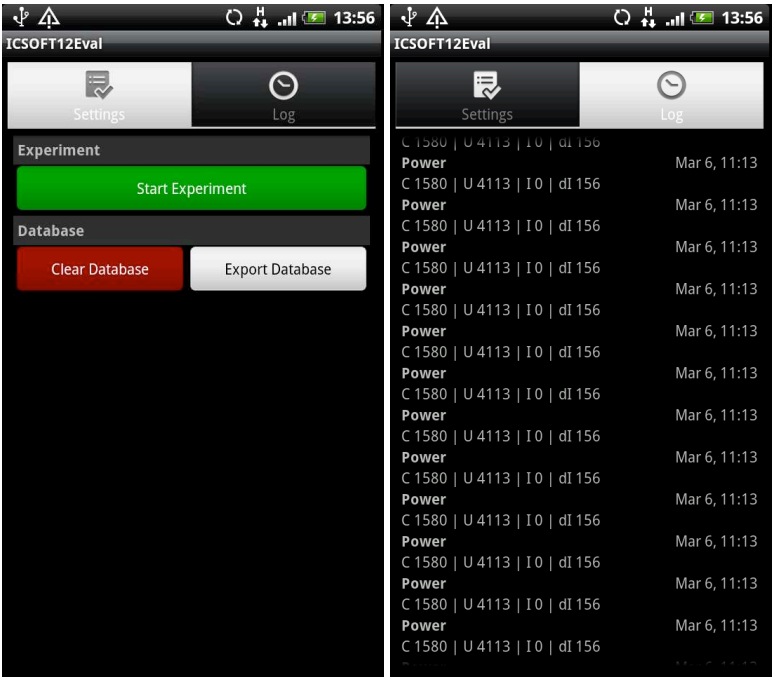
Another important observation is the fact that in contrast to the remaining capacity measurement approach on iOS devices, the second approach can be implemented as a background service on the Android platform. Thereby, gathered data contains less noise by consumers such as the device's display (which has to be kept turned on when measuring on the iOS platform).

5 EVALUATION

We conducted an evaluation to assess the expressiveness of either measuring remaining battery capacity, or measuring battery voltage and discharge current in order to gather energy data on smartphones. The main question to this study was: how does the more simple approach 1 (remaining capacity measurement) perform compared to the more complex approach 2 (voltage and discharge current measurement)? Furthermore, we hypothesized that the first approach is well suited for estimating the total energy demand, while the second approach provides a more granular insight to the energy demand characteristics of a device over time.

5.1 Experimental Setup

The experiment followed the setup of our previous study [HSB12], where we compared a hardware-based measurement approach to our software-based setup. This time, we used an HTC Desire smartphone, because it provides both remaining capacity data, as well as voltage and discharge current. The evaluation software consists of a broad range of algorithms that are executed with increasing runtime and space complexity. Each run took about 20 minutes. During the runs, we measured remaining capacity, voltage, and discharge current with a frequency of 1 Hz. Please refer to [HSB12] for a more detailed description of our experiment application and general setup.



(a) Settings Screen (b) Log Screen

Figure 3: Screenshots of our evaluation app.

Figure 3 presents an overview of our minimalistic evaluation app. The graphical user interface consists of a main settings screen, and a log screen. The experiments and the logging are conducted in background services.

We conducted several measurement runs and detected only very slight deviations between the gathered data.

5.2 Results

The results of our experiment indicate that for a long-term measurement, the total energy demand can be estimated with the remaining capacity approach with high accuracy. However, either the theoretical voltage of the battery, or a real (measured) average voltage over the time of the experiment is required to calculate the energy demand from the capacity difference. In our case, this data was available. During an exemplary 20-minute run (cf. Table 2), we computed a difference of 88 mAh in remaining battery capacity (first logged value: 593 mAh, last logged value: 505 mAh).

Metric	Value
time passed	20.2 min
average electrical power	0.97 W
average voltage	3.709 V
total electrical energy	1174.44 J
difference in remaining capacity	88 mAh
difference in remaining capacity	17.43 %
conversion energy → capacity	87.96 mAh
conversion capacity → energy	1174.96 J

Table 2: Summary of results for one of the test runs.

From this difference, it is easy to convert from the battery's difference in capacity (i. e. electrical charge) to electrical energy by multiplying with the gathered average voltage during the test run. Typically, the battery's voltage remains stable across a discharge cycle, but drops significantly when it is depleted. The 3.709 V measured for the average voltage is very close to the official 3.7 V from the battery's specification. The resulting energy value of 1174.96 J is very close to the 1174.44 J that we measured by integrating over all multiplied voltage and current values. It is also possible to calculate from energy back to capacity by dividing the computed energy value by the average voltage. The resulting 87.96 mAh value is basically identical to the original difference in capacity.

As introduced before, the remaining capacity approach is probably sufficient for long-term total energy demand evaluations. A suitable application scenario would be different kinds of data processing algorithms that are compared over longer periods of time during real-world usage. With the gathered data, it would be possible to identify which one of the compared algorithms performs better in terms of lower energy demand.

However, as indicated by Figure 1 and Figure 4, it is nearly impossible to analyse the exact energetic behaviour of the algorithms, for example in order to identify repeating prolonged use of expensive hardware components. The decrease in remaining capacity is quite steady and makes it hard to identify such patterns. In contrast, the second approach allows the analysis of power consumption over time, which provides more insight to the characteristics.

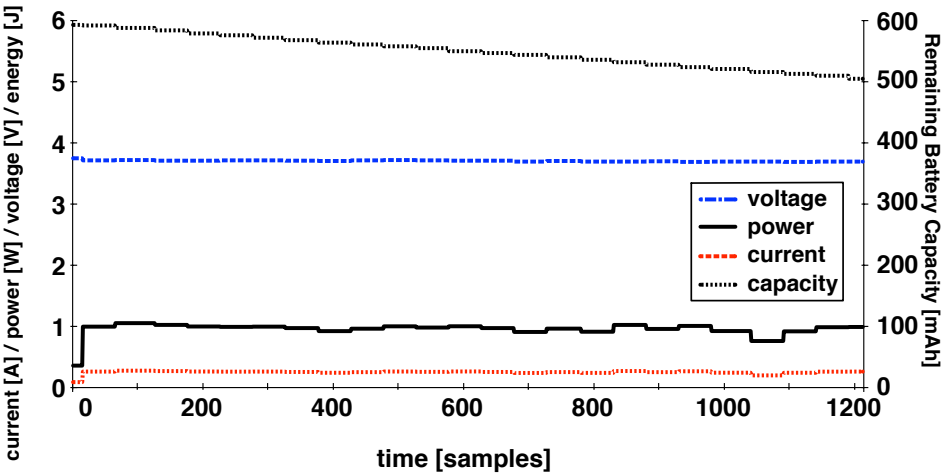


Figure 4: Gathered energy data.

In Figure 1, individual peaks in power consumption are clearly visible and in this case indicate the use of the smartphone's GPS hardware components. In our original experiment, the power comsumption remains stable throughout the measurement period. It is a typical example where the exact progression of energy use is not the most important factor during an evaluation.

6 SUMMARY AND OUTLOOK

In this paper, we presented an overview of software-based energy measurement approaches for smartphones, and a case study to compare these approaches. First, we introduced the remaining capacity-based approach that collects and computes energy data in the form of remaining electrical charge in the device's battery, as approximated by the device's power management unit. Second, we introduced the voltage and discharge current-based approach that collects the device battery's voltage and electrical discharge current. Both sets of values can then be used to derive the temporal progression of energy consumption, as well as the total energy demand of the device during the evaluation period.

We conducted a case study in order to evaluate if both approaches provide sufficient data to compare the energy demand of smartphone applications and hardware use. Our results

indicate that both approaches work equally well for measuring the total demand, but the first approach is not expressive enough to reconstruct the temporal progression of power consumption during an experiment. This is where the second approach is advisable.

In the future, we plan to aggregate our experiences with different software- and hardware-based approaches for measuring the energy requirements of smartphones into a more formal model that allows others to easily select the appropriate measurement approach for a given application and problem space.

References

[BH08] Christian Bunse and Hagen Höpfner. Resource substitution with components — Optimizing Energy Consumption. In José Cordeiro, Boris Shishkov, Alpesh Kumar Ranchordas, and Markus Helfert, editors, *Proceedings of the 3rd International Conference on Software and Data Technologie (ICSOFT 2008), July 5-8, 2008, Porto, Portugal*, volume SE/GSDCA/MUSE, pages 28–35, Setúbal, Portugal, July 2008. INSTICC, INSTICC press. http://hoepfner.ws/images/papers/icsoft08_bunse_hoepfner.pdf.

[BH11] Christian Bunse and Hagen Höpfner. Analyse des Zusammenhangs zwischen Energiebedarf, Dienstgüte und Performanz bei der Ressourcensubstitution in Softwaresystemen. In Wolfgang A. Halang, editor, *Herausforderungen durch Echtzeitbetrieb — Proceedings of the real-time 2011 workshop (Echtzeit 2011); November 3-4, 2011, Boppard, Germany*, Informatik aktuell, pages 101–110, Berlin / Heidelberg, November 2011. Springer. in German, http://dx.doi.org/10.1007/978-3-642-24658-6_12.

[BH12] Christian Bunse and Hagen Höpfner. OCEMES: Measuring Overall and Component-based Energy Demands of Mobile and Embedded System. In *Proceedings of the 42. annual conference of the German computer society (Gesellschaft für Informatik e.V. (GI))*, Lecture Notes in Informatics (LNI), Bonn, Germany, 2012. accepted for publication, forthcoming.

[BHRM11] Christian Bunse, Hagen Höpfner, Suman Roychoudhury, and Essam Mansour. Energy efficient data sorting using standard sorting algorithms. In José Cordeiro, Alpesh Ranchordas, and Boris Shishkov, editors, *Software and Data Technologies — 4th International Conference, ICSOFT 2009, Sofia, Bulgaria, July 26-29, 2009. Revised Selected Papers*, volume 50 of *Communications in Computer and Information Science*, pages 247–260, Berlin / Heidelberg, 2011. Springer. http://dx.doi.org/10.1007/978-3-642-20116-5_19.

[BMSS10] Andreas Beckmann, Ulrich Meyer, Peter Sanders, and Johannes Singler. Energy-efficient sorting using solid state disks. In *Proceedings of the 2010 International Green Computing Conference*, pages 191–202. IEEE, 2010.

[GP10] Laurence Goasduff and Christy Pettey. Gartner Says Worldwide Mobile Device Sales Grew 13.8 Percent in Second Quarter of 2010, But Competition Drove Prices Down. Press Release (online), August 2010. http://www.gartner.com/it/page.jsp?id=1421013.

[GP12] Laurence Goasduff and Christy Pettey. Gartner Says Worldwide Smartphone Sales Soared in Fourth Quarter of 2011 With 47 Percent Growth. Press Release (online), February 2012. http://www.gartner.com/it/page.jsp?id=1924314.

[HB10] Hagen Höpfner and Christian Bunse. Energy Aware Data Management on AVR Micro Controller Based Systems. *ACM SIGSOFT Software Engineering Notes*, 35(3), May 2010. ISSN: 0163-5948, online only, available at `http://dx.doi.org/10.1145/1764810.1764820`.

[HB11] Hagen Höpfner and Christian Bunse. Energy Awareness Needs a Rethinking in Software Development. In Maria Jose Escalona, Boris Shishkov, and José Cordeiro, editors, *Proceedings of the 6th International Conference on Software and Data Technologies, ICSOFT 2011, Seville, Spain, July 18-21, 2011*, volume 2, pages 294–297, Setúbal, Portugal, July 2011. INSTICC, SciTePress. available at `http://hoepfner.ws/images/papers/ICSOFT11_hoepfner_bunse.pdf`.

[HS12] Hagen Höpfner and Maximilian Schirmer. Energy efficient continuous location determination for pedestrian information systems. In Man Lung Yiu, editor, *Proceedings of the 11th International ACM Workshop on Data Engineering for Wireless and Mobile Access (MobiDE 2012), Scottsdale, Arizona, USA, May 20th, 2012*, pages 58–65, New York, NY, USA, 2012. ACM Press. online only, available at `http://dx.doi.org/10.1145/2258056.2258069`.

[HSB12] Hagen Höpfner, Maximilian Schirmer, and Christian Bunse. On measuring smartphones' software energy requirements. In *Proceedings of the 7th International Conference on Software and Data Technologies, ICSOFT 2012, Rome, Italy, July 24-27, 2012*. INSTICC, SciTePress, 2012. accepted for publication, forthcoming.

[KLGT09] Mikkel Baun Kjærgaard, Jakob Langdal, Torben Godsk, and Thomas Toftkjær. EnTracked: energy-efficient robust position tracking for mobile devices. In *MobiSys'09 Proceedings of the 7th international conference on Mobile systems, applications, and services, Wroclaw, Poland, June 22-25, 200*, pages 221–234, New York, NY, USA, 2009. ACM Press. `http://dx.doi.org/10.1145/1555816.1555839`.

[Pow11] PowerTutor. A Power Monitor for Android-Based Mobile Platforms, October 2011. `http://ziyang.eecs.umich.edu/projects/powertutor/index.html`, last access: June 20, 2012.

[SH11] Maximilian Schirmer and Hagen Höpfner. SenST: Approaches for Reducing the Energy Consumption of Smartphone-Based Context Recognition. In Michael Beigl, Henning Christiansen, Thomas Roth-Berghofer, Anders Kofod-Petersen, Kenny R. Coventry, and Hedda Rahel Schmidtke, editors, *Proceedings of the 7th International and Interdisciplinary Conference on Modeling and Using Context 2011 (CONTEXT'11); September 26-30, 2011, Karlsruhe, Germany*, volume 6967 of *Lecture Notes in Computer Science*, pages 250–263, Berlin / Heidelberg, 2011. Springer. `http://dx.doi.org/10.1007/978-3-642-24279-3_27`.

[VOH+04] Jari Veijalainen, Eetu Ojanen, Mohammad Aminul Haq, Ville-Pekka Vahteala, and Mitsuji Matsumoto. Energy Consumption Tradeoffs for Compressed Wireless Data at a Mobile Terminal. *IEICE Transactions on Communications*, E87-B(5):1123–1130, May 2004. `http://search.ieice.org/bin/summary.php?id=e87-b_5_1123`.

[ZTQ+10] Lide Zhang, Birjodh Tiwana, Zhiyun Qian, Zhaoguang Wang, Robert P. Dick, Z. Morley Mao, and Lei Yang. Accurate Online Power Estimation and Automatic Battery Behavior Based Power Model Generation for Smartphones. In *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis, CODES/ISSS 2010, Scottsdale, Arizona, USA, October 24–29, 2010*, pages 105–114, New York, NY, USA, 2010. IEEE/ACM/IFIP, ACM. `http://dx.doi.org/10.1145/1878961.1878982`.