Demo: User Support for Power Management of Continuous Sensing Applications

Chulhong Min[1], Chungkuk Yoo[1], Sangwon Choi[2], Pillsoon Park[3], Seungchul Lee[1], Changhun Lee[1], Seungpyo Choi[1], Seungwoo Kang[4], Youngki Lee[5], Inseok Hwang[6], Younghyun Ju[7], Junehwa Song[1]

[1]School of Computing, KAIST, [2]Information and Electronics Research Institute, KAIST, [3]Division of Web Science Technology, KAIST, [4]Computer Science and Engineering, KOREATECH, [5]School of Information Systems, Singapore Management University, [6]IBM Research – Austin, [7]Naver Labs

[1,3]{chulhong, ckyoo, spchoi, pillsoon.park, seungchul, changhun, spchoi, junesong}@nclab.kaist.ac.kr, [2]sangwonc@kaist.ac.kr, [4]swkang@koreatech.ac.kr, [5]youngkilee@smu.edu.sg, [6]ihwang@us.ibm.com, [7]younghyun.ju@navercorp.com

ABSTRACT

Recently, a number of continuous sensing applications have been actively proposed in research communities and commercially released in the market. However, due to their unique power characteristics, user behavior-dependent battery drain, they bring new challenges for users' power management on these applications. In this demonstration, we present a comprehensive approach to support users' power management for continuous sensing applications. First, at pre-installation time, we provide an instant, personalized power estimation of a continuous sensing application. Without exhaustive trial and error, users can decide judiciously to install a certain application or not. Second, at runtime, we provide mobility-aware battery information. With this information, users can better estimate the phone's remaining battery life based on their imminent mobility conditions and take necessary actions in advance such as carrying an additional battery or minimizing the use of applications.

Categories and Subject Descriptors

C.3 [**Special-Purpose and Application-based Systems**]: Real-time and embedded systems; H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

Keywords

Smartphone; continuous sensing application; battery management; user support

1. INTRODUCTION

Recently, a variety of continuous sensing applications (hereinafter 'apps') have been actively proposed in research communities and commercially released in the market. In the background, they continuously monitor diverse user contexts such as location, physical activity, and conversation [1][4] and provide what users need right on time and place. To save energy for continuous monitoring, they often adopt context-dependent sensing pipelines. The key idea is to trigger high-power sensors selectively by low-power sensors. A common example is a location tracking app. It triggers GPS only when a user's motion is detected by power-efficient accelerometers [4]. Due to such context-dependent logic, the actual power use of continuous sensing apps largely depends on a user's physical behaviors and environmental conditions.

The contextual power consumption of continuous sensing apps brings new challenges for users' power management. First, the power impact of a continuous sensing app largely varies across different users. Thus, it is important to provide a personalized power estimation in order to help users understand the power impact of a sensing app. Second, even for an individual user, the battery drain of a sensing app depends on a user's imminent situation. Thus, while using sensing apps, users can be embarrassed due to unexpected, sudden battery drop.

In this demonstration, we present a comprehensive approach to help sensing app users manage their phone's battery. First, at pre-installation time, we provide an instant, personalized power estimation of a continuous sensing app [2]. Users can decide judiciously to install a certain app without exhaustive trial and error. Second, at runtime, we provide mobility-aware battery information. With this information, users can better estimate the phone's battery life based on their imminent mobility conditions and take necessary actions in advance such as carrying an additional battery or minimizing the use of applications.

2. POWER-IMPACT ESTIMATION AT PRE-INSTALLATION TIME

Providing an accurate, personalized power estimation of a continuous sensing app at pre-installation time is a non-trivial task. A trivial solution may involve computing the average power impact in certain, common use cases. Such estimations, however, would be inaccurate for individual users, since the power use of a sensing app varies noticeably depending on the users' physical activities, phone usage, and other environmental factors.

To take these factors into account, we devise a user-behavior aware power emulation approach. The key idea is to capture the user's physical activities and any other environmental factors, recreate this environment on a cloud-side emulator, and execute the target sensing app on the emulator to track its power use. This approach has three advantages. (1) It provides an individualized power estimation of a continuous sensing app,

SenSys '15, November 1-4, 2015, Seoul, South Korea
ACM 978-1-4503-3631-4/15/11.
DOI: http://dx.doi.org/10.1145/2809695.2817870.

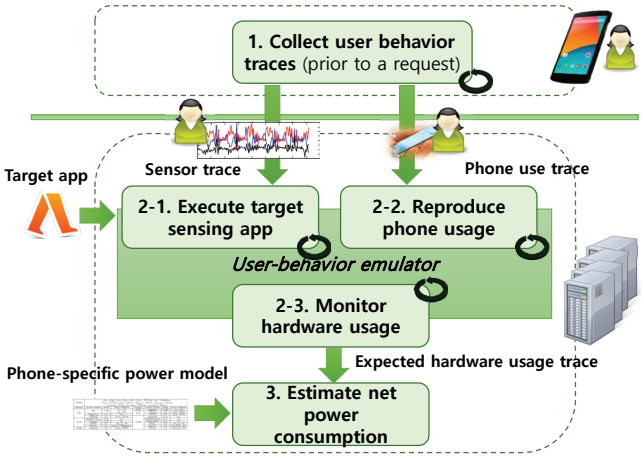

Figure 1. Overall operation for power-impact estimation

since it utilizes the user's physical sensor and phone usage data to construct an emulator environment identical to the user's personal device. (2) It estimates power use of an app without prior power profiling or knowledge of its internal logic. The emulator only requires the app's executable in order to track its hardware use, with no need of its source code. (3) It considers shared hardware use with existing apps by reproducing their resource use.

The system takes three inputs in order to provide an accurate, personalized power impact estimation of a target sensing app: a user's sensor trace, phone use trace, and the sensing app's executable. It outputs the net power increase (mW) due to the target app. The system consists of two major components, a mobile-side trace collector and a cloud-side power emulator, as depicted in Figure 1. The mobile-side trace collector runs on the user's device and is responsible for collecting the user's behavior trace: sensor traces and phone use traces. These traces are then uploaded and managed on the cloud server, which upon a power estimation request, executes its component emulator. The emulator recreates the user's environment from behavior traces, executes the target sensing app, and monitors its hardware use to provide an estimation of its power impact.

3. MONITORING OF MOBILITY-AWARE BATTERY DRAIN

We present mobility-aware battery drain information of running sensing apps in two aspects. First, we provide expected battery life for a set of mobility conditions. Figure 2(right) shows expected battery life at specific mobility conditions. It also displays the expected battery level after a specified amount of time. When the battery level is relatively high, but a long-term activity is expected, e.g., 2-hour driving, users can predict their future battery status depending on their imminent mobility condition.

Second, we provide a historical battery use summary. Figure 2(left) shows the time spent and the battery drained in standby state for each mobility conditions. This historical information is similar to the per-app battery usage that Android provides by default. When an unusual day is expected, e.g., a business trip, users can refer to the past days with similar distribution of mobility conditions.

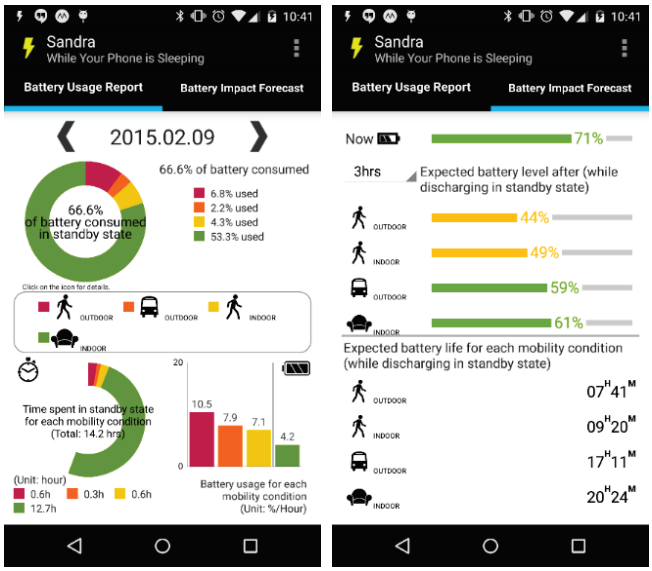

Figure 2. Screenshots of a battery information advisor

4. DEMONSTRATION

We first demonstrate the overall operation of power-impact estimation. As in Figure 1, the key operations are 1) emulating power use of a target app with given user behavior traces, 2) tracking hardware usage statistics generated by the execution of the app on the emulator, and 3) calculating the power consumption based on the hardware usage statistics. To show them effectively, we plan to build web pages and to present a short video. The web pages intuitively show the intermediate results of the internal operations. The video delivers a brief description of the system architecture and operations. We further demonstrate the current prototype of our mobility-aware battery information advisor. We plan to set up a live visualization of mobility-aware battery information, targeting commercial sensing apps. Figure 2 shows the screenshot examples.

5. ACKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. 2011-0018120).

6. REFERENCES

[1] Lee, Y., Min, C., Hwang, C., Lee, J., Hwang, I., Ju, Y., Yoo, C., Moon, M., Lee, U., and Song, J. SocioPhone: Everyday Face-to-face Interaction Monitoring Platform using Multi-phone Sensor Fusion. In *Proc. MobiSys* 2013.

[2] Min, C., Lee, Y., Yoo, C., Kang, S., Choi, S., Park, P., Hwang, I., Ju, Y., Choi, S., and Song, J. PowerForecaster: Predicting Smartphone Power Impact of Continuous Sensing Applications at Pre-installation Time. In *Proc. SenSys* 2015.

[3] Min, C., Yoo, C., Hwang, I., Kang, S., Lee, Y., Lee, S., Park, P., Lee, C., Choi, S., and Song, J. Sandra Helps You Learn: the More You Walk, the More Battery Your Phone Drains. In *Proc. UbiComp* 2015.

[4] Paek, J., Kim, J. and Govindan, R. Energy-Efficient Rate-Adaptive GPS-based Positioning for Smartphones. In *Proc. MobiSys* 2010.