Context Aware Power Management of Mobile Systems for Sensing Applications

Ralf Herrmann
Universität der Bundeswehr München (UniBwM)
Werner-Heisenberg-Weg 39, Munich, Germany
ralf.herrmann@unibw.de

Piero Zappi, Tajana Simunic Rosing
University of California San Diego
9500 Gilman Dr.; La Jolla, CA, 92093
pzappi, tajana@ucsd.edu

ABSTRACT
The widespread adoption of mobile, yet powerful smartphones enable the development of distributed sensing applications. For such applications, the smartphone plays a central role by acting as a bridge between user worn sensors and a remote server. Smartphone energy consumption is a critical issue that should be considered. As the user’s daily context continually changes, the application can dynamically adapt its behavior to the user context to reduce the smartphone energy consumption. In this work, we propose to use context knowledge to dynamically adapt the behavior of sensing applications running on smartphones. In our system a context aware application manager starts, suspends and changes the sampling rate of a sensing application according to current user’s context. We tested our technique on a real system and we show that it can extend smartphone lifetime by 5x.

Categories and Subject Descriptors

General Terms Management, Design, Experimentation

Keywords smart phone, energy, context

1. INTRODUCTION
Today’s smartphones provide enhanced computational, sensing and user interface capabilities that allow the development of complex applications. Smartphones are an effective solution for a number of applications that need continuous sensing. Examples include health care [1], and participatory sensing [2]. These projects focus primarily on data collection, sharing and dissemination aspects. The typical scenario consists of one or more body worn sensors which continuously collect data from the user or the user surrounding. This data is aggregated on the smartphone with the data from its embedded sensors (i.e. GPS, accelerometer etc.). Furthermore, the smartphone acts both as a gateway to a backend server and as interface for data presentation. For example BikeNet [3] is a mobile sensing system used for mapping the cyclist’s experience. In this project a number of sensors are embedded in a bike and communicate with a Nokia N80 phone to aggregate statistics on the user and his behavior. These statistics are sent to a backend server and shared among the user community. Another example is the CitiSense project [6] which relies on body worn air quality sensors and smartphones to provide feedback to users on how air quality affects their health. In these examples a sensing application is continuously running on the smartphone collecting and forwarding data to a backend server. Continuous collect and forward of data from sensors to backend impacts on the smartphone’s battery life especially when the samples needs to be geo tagged. In fact, just tracking location can reduce smartphone lifetime to a few hours [4]. Furthermore, wireless interfaces with the body worn sensors, such as Bluetooth, impact the smartphone battery life. However, often data is collected within a context which is not of interest. For example, in an air quality monitoring application, samples collected within a vehicle fail to detect the real pollutant concentration in a certain area due to the filters in the air condition system. A more efficient approach would rely on user context detection to modify the application behavior as a user moves within different situations.

Context-aware computing refers to a general class of mobile systems that can sense their environment and can adapt their behavior accordingly [14]. For example, in [5] a location aware application running on a PDA provides information relevant to caregivers in a hospital. Energy efficient context recognition has been extensively studied in the past. Related work on context aware computing either focus on energy efficient context recognition techniques [7] or propose energy efficient context management frameworks [8][9]. However, to authors best knowledge, the problem to leverage user’s context to optimize how other applications running on the smartphone utilize system resources has received little attention from the research community. Context knowledge to improve system energy efficiency has been proposed in [10]. Here a stochastic model of the user behavior within a smart home is developed to predict the user’s movements. The user proximity to appliances the system should be executed always by the same device. Furthermore, in [10] only user location information is used, while several other aspects of the user context can be considered. For example, in this work we include user’s mobility and various types of activities.

In this work we propose a context-aware Dynamic Power Management (DPM) subsystem that leverages context knowledge to adapt at run time the behavior of continuous sensing applications. In contrast to previous work on energy efficient context recognition, here we do not propose a novel framework or technique to detect some aspect of context. Rather, we show the benefit of including context knowledge in the management of other applications running on a smartphone. The DPM subsystem can be combined with other energy efficient techniques for context detection to improve system lifetime and scalability.

Our context-aware Dynamic Power Management (DPM) subsystem consists of three components: a Context Recognition...
Service that detects the current user’s context using sensors available in user’s proximity, a Context Manager that manages context detection and issue configuration commands to the Sensing Application which receives the adapt its data collection according to the context dependent commands sent by the Context Manager. For example, if the user is interested in collecting data only in an outdoor area, the Context Managers stops the Sensing Application whenever the user enters a building. Another example is to adjust the sensing rate according to the user mobility to avoid unnecessary sampling over the same area.

We implemented our software architecture on an Android smartphone. We test our context aware DPM using an air quality participatory sensing system. When compared to the case where no context-aware power management is used, our technique increases phone battery lifetime by more than a factor of five while delivering quality results to the users.

2. CONTEXT-AWARE DPM

In this work, we assume that each user carries one or more body worn sensors nodes that collect data from the environment. Sensors nodes communicate with a smartphone that timestamps, geo-tags the data, and forwards it to a server. The context aware DPM manages applications running on the smartphone.

The general architecture of our context-aware DPM is shown in Figure 1. The Context Manager acts as an middle layer between the Sensing Applications and a Context Recognition Service. Each application registers for a set of contexts that are provided by the Context Recognition Service. During this phase each application specify its behavior in each context. For example, an air quality monitoring application is interested in collecting data only when the user is in an outdoor context, or the sampling rate increases when the user is biking with respect to the case when the user is either walking or stationary. The Context Manager start and stops a Sensing Application whenever it detects that the user is within or without a context of interest. The Context Recognition Service is a collection of energy efficient context detection algorithms.

![Figure 1. Context-aware DPM subsystem](image)

Sensing application has three services: sensor sampling, storage and communication. Sensor sampling service communicates with the body area network and retrieves data from the sensors. The storage service saves this data for local processing and forwards it to a server. The Sensing application behavior is controlled by the Context Manager according to current context. For example, sensing may be limited to particular hours of the day or location and the sensing rate may change according to user mobility.

Context Recognition Service is a collection of algorithms to detect different context variables from the available sensors. The Context manager defines which context should be computed and how often according to the application needs. The specific context recognition techniques used by the system are out of the scope of this paper. In this work we implemented our own Context recognition service, however any other energy efficient framework can be used. Note that the context recognition is an extra cost that the system should pay to obtain context knowledge and adapt accordingly. Thus, if a more energy efficient context recognition framework is used, our energy saving improves.

Context manager manages the detection of the context variables and the behavior of the Sensing application. Context variables are evaluated only if an application is registered for it. Whenever an application registers for a context variable it also defines how often that variable should be updated, and the accuracy of the recognition. The Context Manager sets the update period of a context variable to the shortest update period among those requested by each application that registered for that variable. This ensures that the number of context evaluations is minimized. Furthermore, it defines the accuracy in the detection of each variable depending on the needs of each applications. For example, one application may need to know only the area of the city where the user is walking and therefore can use cell tower triangulation, while the other may require a more accurate estimate (i.e. few meters). As long as the second application is running, the Context Manager requests to the Context Recognition Service to provide high accuracy localization using the GPS module. This information is shared to both sensing applications. When the second application ends, the Context Manager relaxes the constraints on the localization accuracy and the Context Recognition Service can rely on other more energy efficient techniques such as cell tower based triangulation.

3. SMARTPHONE POWER MODEL

To simulate the power consumption of the smartphone we developed a power model for the HTC Aria [11] phone we used in our experiments. As the user moves into different contexts the Context Recognition Service and the Context Manager change the status of each component. In the next set of figures, we highlight with red boxes the transitions driven by context changes.

![Figure 2. GPS model with context-aware power controls](image)

GPS operates in three states as shown in Figure 2. The power transitions from OFF/SLEEP to ACTIVE state are enabled when samples are geo-tagged. Similarly, the frequency of GPS sampling is set as a function of the user’s speed – faster when the speed is higher. We do not consider GPS signal strength and number of connected satellites to because that does not sufficiently affect the overall energy consumption [12]. We assume that the GPS module has valid almanac and ephemeris data.

Accelerometer model is shown in Figure 3. Typically, the accelerometer is used to detect user activity. The transition between SLEEP and NORMAL mode is controlled by the application contexts of interest. Since the transition between SLEEP and NORMAL state requires a few ms, the accelerometer is duty cycled between successive context detections.
Figure 3. Accelerometer with context-aware power controls

Wi-Fi model is shown in Figure 4. It operates in either low or high utilization mode depending on the amount of data that needs to be sent. Several examples to efficiently detect available Wi-Fi access points are available in the literature [15]. Wi-Fi is activated and used for communication within contexts where data is sent to a backend server and Wi-Fi network is available. Wi-Fi module is also activated when the user is moving to help with localization.

Figure 4. Wi-Fi model with context-aware power controls

Cellular module is shown in Figure 5. It has three power states: IDLE, CELL_FACH, and CELL_DCH. The cellular module is used to forward data to a backend server and to perform cell tower based localization only when Wi-Fi connectivity is not available since cellular connectivity is less energy efficient [16].

Figure 5. Cellular radio model with context-aware controls

Bluetooth model is shown in Figure 6. Bluetooth is activated when the smartphone collects samples from the sensor board. The module enters in the PARK state between successive sample transmissions. Whenever the user enters in a context of interest, the Context Manager activates the Sensing Application which, scans for available sensors and connects to the sensors of interest. Context dependent application sensing rate modulates the transitions between the PARK and CONNECT.

Figure 6. Bluetooth radio model with context-aware controls

We added models of CPU (Figure 7) and memory (Figure 8). As the application changes its behavior when using the context aware DPM, so the CPU and memory power consumption is adjusted accordingly. CPU’s frequency of operation depends on the level of utilization. When the smartphone operates in different contexts, the applications generate a different number of jobs. Therefore, the job arrival rate and the percentage of time the CPU operates at high or low frequency is set accordingly in our simulations. For the memory, we model a low power mobile 200MHz DDR SDRAM [13].

Figure 7 CPU model with context-aware power controls

For both components we measure the requests inter-arrival time as the phone runs various jobs. We model request inter-arrival times with an exponential distribution. CPU jobs inter-arrival time has been measured on an Xscale ARM processor running at 525 MHz which has performance similar to the processor in the Aria phone. Memory request inter-arrival rate has been obtained using architecture level simulation and memory power measurements.

Figure 8. Memory model with context-aware power controls

4. RESULTS

We implemented and evaluated our context aware DPM within the air quality monitoring Citisense [6] project. Within this project, users carry a sensor node that samples air pollutants and a smartphone that acts as a local aggregator for the data from the sensors, provides connectivity to a backend server and presents feedback to the user regarding his/her exposure (see Figure 9).

The Citisense sensor board is a wearable sensor node able to detect environmental variables (humidity, temperature and barometric pressure) together with a set of pollutants (Nitrogen Dioxide, NO2, and Carbon Monoxide, CO) and to communicate via Bluetooth to the smartphone. The board is able to operate continuously for almost 6 days with the provided 1800mAh Li-ion battery. The smartphone is an Android HTC Aria [11] where our context-aware power management DPM runs.
The Context Recognition Service detects: user’s mobility and location, if the user is indoor/outdoor and if he/she is resting. Here we do not focus on which technique is used to detect the user context. In fact, the energy savings are a consequence of how the application adapts to the current context. The context detection is an extra cost that the system has to pay to gain the context knowledge. By using more energy efficient techniques, we reduce this extra cost, thus increasing the gain in energy saving.

The smartphone accelerometer is used to recognize the following mobility states: walking, riding a bike, going by car or bus and standing. 67 hours of recorded data has been used to train and characterize a 348 tree classifier. We achieved 89.4% accuracy and execution time is 2.2 sec.

Location is provided using area detection. In our experiments indoor and outdoor location is detected based on the network location provider with an accuracy of 10 m to several hundred meters depending on the number of Wi-Fi access points and cell towers in the connection range. Average execution time is 1 sec.

Indoor/outdoor detection depends on the ability to obtain from the GPS module a location fix. When the phone is outside, the GPS localization takes 2 sec. on average. When indoor, the algorithm takes 42ms to execute plus a waiting delay of 10 sec.

We use the magnitude of the acceleration to detect if the user is at rest or moving. Our assumption is that when the user is resting, he/she doesn’t carry the smartphone and leaves it on a still surface (e.g. table). The system assumes that the other context variables remain unchanged when the user is at rest, therefore we stop also the context detection. Detection of this context takes 1 sec.

4.1 Measurement Results

Due to HVAC and air filters, the user exposure to pollutants within buildings and vehicles is negligible. Therefore, we implemented a system where sensing and reporting is performed only if the user is outdoors (either resting or moving) and is not in a vehicle. We compare two implementations of the collaborative sensing application. One case is where no context adaptation is performed (naïve system), and the other uses our context aware DPM. When the application does not leverage user’s context, it periodically samples the sensors and forwards the data to the backend. For both cases keep standard phones energy saving policies (i.e. turn off the screen when idle etc.).

We carried the sensing system for two days. The application is started at midnight and run until the battery is discharged. A map with a subset of measurements is shown in Figure 10. High pollutant locations (in red) occur on main roads. We compare the battery levels with and without context awareness in Figure 11. With our system the increase in battery lifetime is dramatic: from 577 minutes without context-aware power management to 3061 minutes when it is, for an increase of 5.3x. The quality of the user’s exposure estimation is unchanged between the two cases.

A detailed analysis shows that with our context aware DPM the smartphone power consumption is lower at the beginning and the end of the day, when the user is at home and the phone is at rest.

4.2 Simulation results

We simulate the phone power consumption within different scenarios through the finite state machines shown in the previous section. In our experiments the smartphone is used as a gateway to a backend server, so we did not model the LCD power cost, as it was not changed by our implementation.
Figure 13 shows the simulated power consumption of the smartphone components. As expected, GPS is the main source of power consumption followed by the wireless communication modules (Bluetooth, Wi-Fi and Cellular radio). This pushes the use of low power techniques to reduce the overhead in energy consumption associated to these components. Furthermore, local computation should be preferred to reduce the amount of data sent through the wireless channel.

![Smartphone components energy consumption](image)

**Figure 13. Smartphone components energy consumption when the context-aware sensing is used**

Table 2 compares the power consumption of phone components when using our context aware technique with the naïve system that streams data continuously. As can be seen from this table, our context aware technique is able to dramatically reduce the power consumption of several components. Since the Sensing Application stops when the user is indoors, power consumption of Bluetooth and GPS is 86% lower when using our policy. On the other hand, the cellular and Wi-Fi module power consumption do not change much. The reason is that these components are used also to perform user localization in addition to data transfer, so this affects their energy cost. Finally, note that the accelerometer is used only for the context recognition. This is an extra cost the system has to pay to get context knowledge.

<table>
<thead>
<tr>
<th>Component</th>
<th>Context Aware (mW)</th>
<th>Naïve system (mW)</th>
<th>Power savings (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>2.8</td>
<td>37.5</td>
<td>92.4</td>
</tr>
<tr>
<td>Memory</td>
<td>8.5</td>
<td>40.6</td>
<td>79.0</td>
</tr>
<tr>
<td>Accelerometer</td>
<td>0.02</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Wi-Fi</td>
<td>16.8</td>
<td>20.4</td>
<td>17.5</td>
</tr>
<tr>
<td>GPS</td>
<td>32.3</td>
<td>242.2</td>
<td>86.7</td>
</tr>
<tr>
<td>Bluetooth</td>
<td>14.0</td>
<td>103.1</td>
<td>86.4</td>
</tr>
<tr>
<td>Cellular</td>
<td>13.8</td>
<td>17.8</td>
<td>22.6</td>
</tr>
<tr>
<td>Total</td>
<td>85.2</td>
<td>461.7</td>
<td>80.9</td>
</tr>
</tbody>
</table>

**Table 2. Average power consumption per component**

5. CONCLUSION

In this paper, we propose the use of context knowledge to improve energy efficiency of sensing applications running on smartphones. Our context aware DPM is able to recognize and adapt to changing context. We show that with our approach we are able to extend the smartphone lifetime by more than 5x. We also analyze per HW component contributions to energy cost when context awareness is enabled. As expected, highest contributors are associated to GPS and wireless communication modules.

6. ACKNOWLEDGMENTS

This research was supported by NSF Grant CNS-0932403, Center for Networked Systems (http://cns.ucsd.edu) grants CNS08TR and CNS10TR, and Qualcomm.

7. REFERENCES


