Empowering Developers to Estimate App Energy Consumption

Radhika Mittal, UC Berkeley
Aman Kansal & Ranveer Chandra, Microsoft Research
Phone's battery life is critical for performance and user experience. Apps are responsible for a large fraction of energy drain, making it difficult for app developers to estimate energy trade-offs.
Energy Efficient Apps

Developer wants to download and display an image. Which one will consume less energy?

Image1 – 18kB
Communication cost – 2J
Display power – 600mW

Image2 – 1MB
Communication cost – 10J
Display Power – 350mW

What tools do developer have to make the right decision?
# Tools to Estimate App Energy

<table>
<thead>
<tr>
<th>Tools to Estimate App Energy</th>
<th>Ease of use &amp; set-up costs</th>
<th>Component energy break-up</th>
<th>Runs in emulator</th>
<th>Repeatability</th>
<th>What-if analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measurement using Power Meter [Flinn et al]</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Software/Event Profiling based power modeling tools [Pathak et al, Dong et al]</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>
Our Approach: WattsOn

- Enables app developers to estimate energy consumed by different components of a phone, while debugging on emulator

- Allows what-if analysis
Basic Structure of WattsOn

- App Developer’s Code
- Debugging on emulator (WattsOn runs in the background)
- Energy Breakdown

Exploration of varying parameters like:
- network quality
- signal strength
- screen brightness
- mobile operator
- handset
## Tools to Estimate App Energy

<table>
<thead>
<tr>
<th></th>
<th>Ease of use and set-up costs</th>
<th>Component energy break-up</th>
<th>Runs in emulator</th>
<th>Repeatability</th>
<th>What-if analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measurement using Power Meter [Flinn et al]</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Software/Event Profiling based power modeling tools [Pathak et al, Dong et al]</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>WattsOn</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

**MobiCom, 25 Aug 2012**
WattsOn Design
Overview

Device Manufacturer, or OS Provider Labs

Target Mobile Device
  Resource Profiling
  Power Profiling

Power Model Generation

Mobile Phone Emulator
  Resource Scaling
  Hardware Resources

Test Application

Resource Profiling

Developer Workstation

Energy Calculation

APP ENERGY CONSUMPTION
WattsOn
Power Modeling
Resource Scaling

Network

CPU

Display
Challenges

- Emulator runs on PC over Broadband Internet
- Phone uses cellular network (3G, 4G...)

- Cellular energy consumption varies with
  - Signal Strength
  - Network Quality
  - Mobile Operator
  - 3G chipset
Power Modeling

TCP server was setup with Packet Sniffer running on it to capture the packets transmitted to/from a 3G device (phone, chipset dev. board)

Windows Phone client containing Qualcomm 3G chipset

Computer client connected to an Ericsson 3G chipset and attenuator
Observations

Normal conditions (AT&T)

- 3 network states

Congested network (daytime)

- Losses interplay poorly with TCP

Weak signal (1-bar)

- More power needed at weak signal

Another MO (T-Mobile)

- Power consumed varies with MO
Resource Scaling

- **Trace Stretching** –
  - Inconsistency with other resource utilizations which depend on network activities
  - Effect on other parameters of network flow not captured

We perform **link shaping** – NEWT, Linux Traffic Shaper…
Network Emulation

WattsOn
- Set network conditions
- Compute network energy from models

User
- Dev application in emulator

Kernel
- Uploaded Frames
- Scaled Download
- TCP/IP
- Sniffer Device Driver
- Layer 2.5 Traffic Shaper
- NIC Device Driver
- Scaled Upload
- Downloaded Frames

3G Network Parameters

Packet Sniffer
Network Accuracy Testing

Average Error - 4.73% across all tests
WattsOn
Power Modeling
Resource Scaling

CPU
Network
Display
Challenges

- LCD and OLED power models show linear and additive properties

- Most mobile devices today use Active Matrix OLED
  - optimizes power by adjusting screen brightness based on the scene displayed
  - additive and linear properties no longer applicable
AMOLED Observations

Non-linearity at high magnitudes

Dependency on other pixels

Dependency on color
Power Model

- Basic linear model from OLED
- Power optimizations modeled using a look-up table
  - 16×16×16 entries (each of R, G, B discretized to 16 levels)
  - Measurement done when color occupies entire screen

\[ P_{\text{display}} = \theta(s) \cdot L(s) + (1 - \theta(s)) \cdot O(s) \]

- \( \theta(s) \) - controls the fraction of screen area for which power optimizations matter (based on color thresholds)
- \( s \) represents the screenshot displayed
Resource Scaling

- Display model depends on pixel power
  - Number of pixels in emulator screen may vary
Display Accuracy Testing

Testing accuracy with simple colors
Display Accuracy Testing

Testing with 30 images
Display Accuracy Testing

Testing sub-sampling of pixels for optimization

![Bar chart showing error percentage for different pixel subsampling rates.](chart.png)
WattsOn
Power Modeling
Resource Scaling

Network

CPU

Display
Challenges

- Emulator PC has much faster CPU than phone:
  - For e.g. 100% utilization on phone with 1GHz CPU maps to 13.8% utilization 2.7 GHz Intel Core-2 Quad-Core processor
  - Other differences like cache size, front side bus speed, ...

- Scaling cannot be done after capturing data:
  - CPU speed affects activities of other components
  - May lead to imbalance in readings.
Restrict the number of processor cycles available to the emulator.

**WattsOn**
- Control Thread
- Compute CPU energy
  \[ P = \alpha \times u \]

DEVELOPER APPLICATION IN EMULATOR

KERNEL
- Lower priority
- Higher priority

OS APIs (Affinity, Priority)

COMPUTER HARDWARE

CPU 0
- ...
- CPU n

Performance Monitor

MobiCom, 25 Aug 2012
CPU Accuracy Testing

Average Error - 9.73% across all tests
Evaluation of WattsOn
Browser App

MobiCom, 25 Aug 2012
Browser App

Average Error: 4.64%
Case Study
App Design Decisions

What’s the energy overhead for desirable app features:

- **Portability** – Hybrid (HTML5) app or Native?
- **Rich Graphics** vs. simplistic icons
- **Animation**

![Simple icon](image1) 18KB

![Rich graphics](image2) 138KB

Animation (90kB)
Portability and rich graphics has negligible energy overhead

- network energy consumption dominated by tail state
App Design Decisions

Display consumes major fraction of energy usage
• avoiding brighter images leads to significant energy savings
Including animation leads to higher CPU energy consumption.
Conclusion

- WattsOn – lightweight emulation time tool for developers
  - Not biased by network conditions or device configuration
  - Enables what-if analysis
  - Average Energy Error: 4% - 9%

- Future Work
  - Power model for sensors e.g. GPS, accelerometers
  - Energy debugging by linking spikes to code snippets
Thank you!!