Cellular Networks and Mobile Computing
COMS 6998-8, Spring 2012

Instructor: Li Erran Li
(lel2139@columbia.edu)

http://www.cs.columbia.edu/~coms6998-8/

4/9/2012: OS Support for Energy and Sensor Management (or Rethinking Mobile OS)
Rethinking Mobile OS

• What abstraction should mobile OS provide to apps?
  – Should the OS provide fine-grained battery power management?
  – Should the OS provide high level contextual information inferred from low level sensor information?
  – Should social network apps be in the OS? An OS system call to send a tweet is much more efficient.
Energy Management in Mobile Devices with the Cinder Operating System

Arjun Roy, Stephen M. Rumble, Ryan Stutsman, Phil Levis, and David Mazières
Stanford University

Nickolai Zeldovich
MIT CSAIL

Cellular Networks and Mobile Computing
(COMS 6998-8)

4/9/12

Courtesy S. Rumble et. al.
The State of Smartphones

• Smartphones are a dominant computing platform

• Energy is the limiting resource on smartphones

• OSes don’t provide any control over it

Cellular Networks and Mobile Computing (COMS 6998-8)

Courtesy S. Rumble et. al.
Cinder: Rethinking the Mobile OS

- Track application energy consumption

- **Fine-grained** energy control primitives
  - For users
    - Limit application energy consumption
  - For applications
    - Use app specific knowledge to manage energy
    - Even within process boundary
Real-world Implementation

• Runs on the HTC Dream (T-Mobile G1)
• Based on HiStar
• ~15,000 lines of code excluding drivers
• Working
  – Display, keyboard
  – Texting
  – 3G Data Radio
  – Answering/Placing Calls (no audio)
• Also runs on x86_64 desktops/laptops
Existing Approaches

• Android
  – Provides visibility
    • Estimates energy consumption for apps
  – No control

• Prior research (ECOSystem)
  – Similar visibility
  – Simple control
    • Try to meet target battery lifetime on a laptop
Outline

• New Abstractions for Control
• Examples
• The Problem of Closed Platforms
• Cinder-Linux

4/9/12
Cellular Networks and Mobile Computing (COMS 6998-8)

Courtesy S. Rumble et. al.
HiStar Overview

- Make all state explicit, track all communication

---

4/9/12

Cellular Networks and Mobile Computing (COMS 6998-8)
HiStar Overview (Cont’d)

• Narrow kernel interface, few comm. channels
  – Minimal mechanism: enough for a Unix library
  – Strong control over information flow

• Unix support implemented as user-level library
  – Unix communication channels are made explicit, in terms of HiStar's mechanisms
  – Provides control over the gamut of Unix channels
HiStar kernel objects

Container
(Directory)

- Segment
(Data)
- Address Space
- Thread
- Gate
(IPC)

Device
(Network)
HiStar kernel objects

Think of labels as a “tainted” bit

Container (Directory)

Segment (Data)

Address Space

Thread

Gate (IPC)

Label

Label

Label

Label

Device (Network)
Power modeling

- Active research area
- Measure offline in isolation
  - Device states
  - System calls
- Bill threads online using offline analysis
  - CPU
  - Data radio
  - GPS
  - Etc.
Energy Control Policies

• **Reserves** limit *quantity* of energy use
  – Subdivision and isolation between apps

• **Taps** limit *rate* of energy use
  – Enforces “lifetime” type policies
Reserves

15 kJ
Reserves

• Virtualized batteries
  – Subdivide energy
Reserves

- Virtualized batteries
  - Subdivide energy

15 kJ

10 kJ

5 kJ

Courtesy S. Rumble et. al.
Reserves

- All threads run in the context of a reserve
  - Accounting
  - Control

15 kJ

10 kJ

5 kJ

Browser

4/9/12

Cellular Networks and Mobile Computing (COMS 6998-8)

Courtesy S. Rumble et. al.
Reserves

• All threads run in the context of a reserve
  – Accounting
  – Control

15 kJ

10 kJ

2 kJ

Browser

Courtesy S. Rumble et. al.
Reserves

• All threads run in the context of a reserve
  – Accounting
  – Control

• De-schedule threads with exhausted reserves
Taps

• A rate transfer between reserves

15 kJ
700 mW

Browser

Courtesy S. Rumble et. al.
Taps

- A rate transfer between reserves

Courtesy S. Rumble et. al.
Taps

• A rate transfer between reserves
• Allows hierarchies
Taps

• A rate transfer between reserves
• Allows hierarchies
Taps

- A rate transfer between reserves
- Allows hierarchies
Taps

• A rate transfer between reserves
• Allows hierarchies
Taps

• A rate transfer between reserves
• Allows hierarchies
• Backward taps prevent hoarding
Battery

15 kJ
Consumption

15 kJ

Browser

Courtesy S. Rumble et. al.
Throttling

• Throttle consumption
  – Taps

Browser

15 kJ
700 mW

Courtesy S. Rumble et. al.
Accommodate Bursty Apps

• Throttle consumption
  – Taps

• Allow bursty apps
  – Reserves

Browser

700 mW
15 kJ

Courtesy S. Rumble et. al.
Accommodate Bursty Apps

• Throttle consumption
  – Taps
• Allow bursty apps
  – Reserves

15 kJ
700 mW

Browser

Courtesy S. Rumble et. al.
Accommodate Bursty Apps

• Throttle consumption
  – Taps
• Allow bursty apps
  – Reserves

15 kJ
700 mW

Courtesy S. Rumble et. al.
Accommodate Bursty Apps

- Throttle consumption
  - Taps
- Allow bursty apps
  - Reserves

Courtesy S. Rumble et. al.
Accommodate Bursty Apps

- Throttle consumption
  - Taps
- Allow bursty apps
  - Reserves

4/9/12
Accommodate Bursty Apps

• Throttle consumption
  – Taps
• Allow bursty apps
  – Reserves

Courtesy S. Rumble et. al.
Prevent Hoarding

- Throttle consumption
  - Taps
- Allow bursty apps
  - Reserves
- Prevent hoarding
  - Backward Taps

Browser

15 kJ

700 mW

10%/sec

Cellular Networks and Mobile Computing (COMS 6998-8)

Courtesy S. Rumble et. al.
Fine-grained App Control

- Throttle consumption
  - Taps
- Allow bursty apps
  - Reserves
- Prevent hoarding
  - Backward Taps
- Fine-grained, app specific policies
  - Nesting
Subdivision & Isolation

- “Virtualized batteries”
- Guarantee energy to certain apps

15 kJ
1 kJ
14 kJ

911
Browser

Cellular Networks and Mobile Computing (COMS 6998-8)

39

Courtesy S. Rumble et. al.
Outline

• New Abstractions for Control
• Examples
• The Problem of Closed Platforms
• Cinder-Linux
ewrap

ewrap 60 ./spin

15 kJ

60 mW

spin
ewrap

```
100 bash -c "
    ewrap 60 ./spin &
    ewrap 70 ./spin &"
```
Background Apps

• Meet users’ expectations

• Apps they can’t see on screen shouldn’t cause battery drain
Background Apps

• Meet users’ expectations

• Apps they can’t see on screen shouldn’t cause battery drain
Background Apps

• Meet users’ expectations

• Apps they can’t see on screen shouldn’t cause battery drain
Outline

• New Abstractions for Control
• Examples
• The Problem of Closed Platforms
• Cinder-Linux
HTC Dream Design

• ARM11 CPU
  – Runs applications
• “Secure” ARM9
  – Controls devices

• Binary blob shared object to interact

• Difficult to reverse engineer the protocol
Developing for a Mobile Phone

• Getting worse
  – Locked bootloaders, software integrity checks
  – Research OSes on phones at all in the future?

• Evolution allowed, but not revolution
  – Can tweak apps, OS; cannot replace OS

• Dominant user computing platform
  – Systems research community is locked out
Outline

• New Abstractions for Control
• Examples
• The Problem of Closed Platforms
• Cinder-Linux
Cinder-Linux

• Implemented reserves and taps on Linux
  – Source code:
    http://www.scs.stanford.edu/histar/src

• Easier access to devices

• Allow experimentation with more sophisticated workloads
The Problem with IPC

• Apps request service from daemons
• Daemons do work for the app
  – OS must bill the app correctly

• Not a problem on Cinder because of “gates”
Gates

• The basis for IPC
• A named entry point in an address space
• To request service, client threads enter the address space of a server
• Simplifies tracking
Cinder-Linux

• Must augment all IPCs across processes
  – Logic scattered throughout userspace
Conclusions

+ Users can control app energy use
+ Apps can leverage developer knowledge
+ Energy tracking works across boundaries
+ Easy to write “energy aware” applications

- Fighting manufacturer lock down
- Many knobs, may require sophistication
What’s Next?

• Incorporate sophisticated energy modeling research, e.g. eprof

• Abstractions on Linux
  – Augment userspace code for energy tracking or do we need it?
    • eprof states most energy are consumed for accessing peripheral devices

• Android on Cinder
Mobile Apps: It’s Time to Move Up to ConDOS

ConDOS: the Context Dataflow OS

david chu• aman kansal• jie liu• feng zhao•
microsoft research redmond• microsoft research asia•
How Might New Apps Use New Sensors?

SENSORS
- camera
- microphone (x2)
- magnetometer
- accelerometers
- NFC
- infrared camera
- light sensors
- barometer
- GPS
- camera (x2)
- health sensors
- thermometer

Cellular Networks and Mobile Computing (COMS 6998-8)
**OfficeFit app**

- contextual fitness reminders in the office
  - “Don’t slouch while sitting.”
  - “You’ve been at your desk for too long.”
  - “Take the stairs instead of the elevator.”

- how it works
  - motion from IMU + sound from mic → various fitness activities
  - do this continuously
context data from sensors

• key pieces are ready
  – sensor hardware
  – application scenarios
  – algorithms (high accuracy inference, signal processing, db, etc.)

• where is the context?

• who is responsible for context?
  – individual apps
    • ... but mobile OSs limit apps to foreground
    • ... or apps can run anything in the background(!)
  – the cloud
    • ... but energy cost of TX/RX is high
  – the mobile OS

Motion State  Logical Location  Interruptible
sitting, walking, running home, office, mall  yes, no

Cellular Networks and Mobile Computing (COMS 6998-8)

Courtesy D. Chu et. al.
ConDOS design proposal

• export *Context Data Units (CDUs)* rather than raw sensor data
  – higher-level abstraction than bytes
  – apps query or subscribe to CDUs

• each CDU is defined by a CDU Generator: a graph of processing components
  – combine Generators into composite context dataflow (like packet dataflow [kohler ’00])
  – provide a base CDU vocabulary (that is extensible)
benefits of OS-managed context

1. System services can use context

Kernel Services

- Scheduling
- Memory Management
- I/O
- Security
- Energy Management

Apps

Contextual Data Units

Hardware Abstraction Layer
system services can use context

• memory management
  – preload calendar, email when *in the office*

<table>
<thead>
<tr>
<th>Context</th>
<th>Preload Action</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>in the office</em></td>
<td>Email, Calendar</td>
</tr>
<tr>
<td><em>at a party</em></td>
<td>Twitter, Facebook</td>
</tr>
</tbody>
</table>

• I/O
  – ring volume adjusted based on *conversation*
  – networking params dictated by *movement* [Balakrishnan ’10]
system services can use context

• security
  – auto password unlock when at *home*
  – lend your phone to others easily [liu ’09]

• energy management
  – predict time-to-recharge based on context
benefits of OS-managed context

2. Privacy enforced by OS protection
better sensor privacy

- mobile privacy is under attack [TaintDroid]
  - protecting raw sensor data is “trust the EULA”
  - 2/3 of popular apps use your data suspiciously
  - no idea what your raw data might be used for

- OS-managed context lets us do better
  - app install time: per CDU type access control
    - ... vs. per sensor type access control
  - app run time: visual inspection of CDUs shared [Howell ’10]
    - ... vs. no comprehension of what is being shared
  - enforcement is low overhead
toward a design

other OS services
- Scheduling
- I/O
- Memory Management
- Security
- Energy Management

context dataflow example

CDU1
- Logical Location
  - home, office, mall
- Location DB
- Geolocation
  - GPS, Cell, WiFi

CDU2
- Motion State
  - sitting, walking, running
- Motion Features
- IMU
  - accel, gyro, mag

CDU3
- Interruptible
  - yes, no
- Audio Features
- Audio
- Silence Filter

User space
- Logical Location
- Motion State
- Audio Features

Kernel space
- Location DB
- Motion Features
- Silence Filter

app A
app G
app Z

User space

Kernel space

4/9/12

Cellular Networks and Mobile Computing
(COMS 6998-8)

Courtesy D. Chu et. al.
• mobile OSs that don’t make sense make no sense
  • ConDOS offers context as a primary app-OS interface
• apps, OS services and User Privacy may all benefit
Closing Thoughts

• Reserve and tap in Cinder enable fine grained control on application energy consumption.
  – But there is no easy answer on when to use them?

• When to generate contextual information?
  – A high overhead is incurred to acquire it.

• What else need OS support?