Amulet: An Energy-Efficient, Multi-Application Wearable Platform

Josiah Hester*, Travis Peters†, Tianlong Yun†, Ronald Peterson†, Joseph Skinner†, Bhargav Golla*, Kevin Storer*, Steven Hearndon*, Sarah Lord†, Ryan Halter†, David Kotz†, Jacob Sorber*

*Clemson University and †Dartmouth College
Wearables

Wristbands
Long lived wearables usually for fitness sensing, with longer lifetimes, but closed source and hardware.

Smartwatches
Very flexible development platforms, with short lifetime, often closed operating systems and hardware.
Tradeoffs

Wristbands

Pros:
- Long lifetime

Cons:
- Closed platform
- Not flexible

Smartwatches

Pros:
- Flexibility

Cons:
- Closed hardware or software
- Short lifetimes
Shortcomings

Flexibility

• Not open

Lifetime

• Does not enable mHealth
mHealth
mobile systems in healthcare

Courtesy: http://sarahcatt.blogspot.com/

Courtesy: experientia.com

Courtesy: sana.mit.edu

Courtesy: seniorcarecenter.com

Courtesy: good.is
Amulet Platform
This Talk

1. Design
2. Implementation
3. Evaluation
Design

Multiple applications
  • Need app isolation

Long battery lifetime
  • Designed for low power operation

Developer tools
  • Focused on energy

Usable

Open Source and Hardware
Multi Application

Flexibility

- Multiple developers, multiple apps
- Users have different needs

Security

- Sandboxing for isolation among apps
- Access control for sensors, peripherals
Lifetime

Need weeks and months

- Not hours and days!
- Support long term studies, deployments
1. Where is my energy going?

2. How does the environment, and the user behavior change energy?

3. What can I change in my code to increase the lifetime?
Open

Open Source
- Use, adapt, change

Open Hardware
- Remix, redo, enhance
Amulet

1. Open Hardware Wearable
2. Amulet-OS and API
3. Amulet Firmware Toolchain
4. Amulet Resource Profiler and UI
Device

Sensors
- 3-axis gyroscope, ST Electronics L3GD20H
- 3-axis nano-power accelerometer, Analog ADXL362
- Ambient light, UVA/B, temp, sound, battery

Computing
- Nordic nRF51822, ARM Cortex M0, 32K RAM, 256K FLASH
- TI MSP430FR5989, 2KB SRAM, 128KB FRAM
- microSD card slot

Network
- BLE radio (Central & Peripheral)
- Supported protocols: heartrate, battery, running services

Output
- Monochrome 128x128 Sharp Memory LCD
- or two single color LEDs
- haptic feedback via vibrator motor

Input
- two buttons
- capacitive touch slider
- accelerometer

Battery
- Polymer Li-Ion, 110 mAh, 3.7V, MCP73831 recharge
Amulet-OS

Apps: finite-state machines w/memory

- set of states, variables, and event handlers
- all state is explicit, in non-volatile storage
- no threads: handlers run to completion
- API calls post event to relevant system service
Subscribe to sensors, log data, communicate, interact.
Amulet Firmware Toolchain

Firmware analysis, translation, compile

- Manage multiple applications
- Analyze for isolation
- Profile for energy and memory usage

App Isolation and Resource Profiling
AFT Workflow

Some restrictions on C:
1. no dynamic memory
2. no pointers
3. no recursion

Check app access control and language violations.
AFT Workflow

1. Step 1: Verify compliance with Amulet C. Insert runtime checks (array bounds, access control).
2. Step 2: Analyze resource usage.
3. Step 3: Visualize usage; Merge apps.
4. Step 4: Compile and link with the runtime system and libraries.
5. Step 5: Install image on device.

Profile energy and memory resource usage.
AFT Workflow

1. Verify compliance with Amulet C and insert runtime checks (array bounds, access control)
2. Analyze resource usage
3. Visualize usage and merge apps
4. Compile and link with the runtime system and libraries
5. Install image on device

Diagram:
- Analyzer & Translator
- Resource Profiler
- App Merger
- Compiler
- Linker
- Installer
- ARP-View
- AmuletOS
- Libraries
- Custom code
- Existing tool
- QM File with Amulet C
- C code
- Energy model
- Device profile
Amulet Resource Profiler (ARP)

Step 1: Verify compliance with Amulet C. Insert runtime checks (array bounds, access control)

Step 2: Analyze resource usage

Step 3: Visualize usage; Merge apps

Step 4: Compile and link with the runtime system and libraries

Step 5: Install image on device
Amulet Resource Profiler

Designing for low power is not enough

- Developers can always write bad apps
- This can be because of ignorance
- Or because tools don't exist!

Must support developers!
89% of users consider battery lifetime the most important feature[1].

Energy is a first class concern for users.

Why not for developers?

Resource Model

Concerned with energy

- Secondary concern: memory

Model the device itself

- One time, at device manufacture

Model the application(s)

- Compile time
Device Energy Model

**Device Profile**
- Hardware Info
  - Steady state draw
  - Sleep currents
  - Sensor power costs
  - Device memory
  - API Calls

Generated once per device type.
// Update temperature if changed:
uint8_t new_temp = AmuletGetTemperature();

if (new_temp != temp) {
    temp = new_temp;
    char temp_disp[5];
    AmuletITOA(temp, temp_disp);
    char F[2] = "F";
    AmuletConcat(temp_disp, F);
    AmuletClearRect(0, 75, LCD_HORIZONTAL_MAX, MEDIUM_FONT_SIZE);
    AmuletMediumCenteredText(75, temp_disp);
    refresh_display = 1;
}
ARP-View

Interface for energy insights

- Generated at compile time

Can model user behavior

- Model environment triggers as well

Explore design tradeoffs
ARP-View

Energy focused development.
Evaluation

1. Battery Lifetime

Battery lifetimes ranging from 2 weeks, to 8 months.
Evaluation

ARP Prediction Accuracy

Prediction accuracy of 90-98% for our apps.
User Study

ARP-View: usability and energy

- what is the developer energy mental model?
- does ARP-view force devs to think about energy?
- 10 programmers, 30 minute task and survey
- 9/10 subject reported positive outcomes

ARP-view assists developers.
Pilot Study

mHealth: smoking cessation surveys

- monitored heart rate using BLE
- presented surveys at intervals during the day
- recorded survey responses and heart rate
- 6 participants, 1 week, 48 hours of deployment
- usability survey that informed hardware rev(s)

Demonstrates mHealth feasibility.
Evaluation

Great battery lifetimes
  • 2 weeks to 8 months

Accurate prediction results
  • 90-98% accuracy

Usable by users, researchers, and developers.
Future

Secure firmware toolchain

- OTA firmware updates and security

Body are health network (BAHN)

- Extending the reach of Amulet

Energy Harvesting Wearables

- Solar, or vibration powered, no charging!

Priority: Enabling your applications!
Summary

Amulet is...

1. Open source, open hardware, multi-app wearable device
2. Firmware toolchain isolating applications and resource profiling.
3. Energy focused application development with ARP-View

amulet-project.org

https://github.com/AmuletGroup/amulet-project