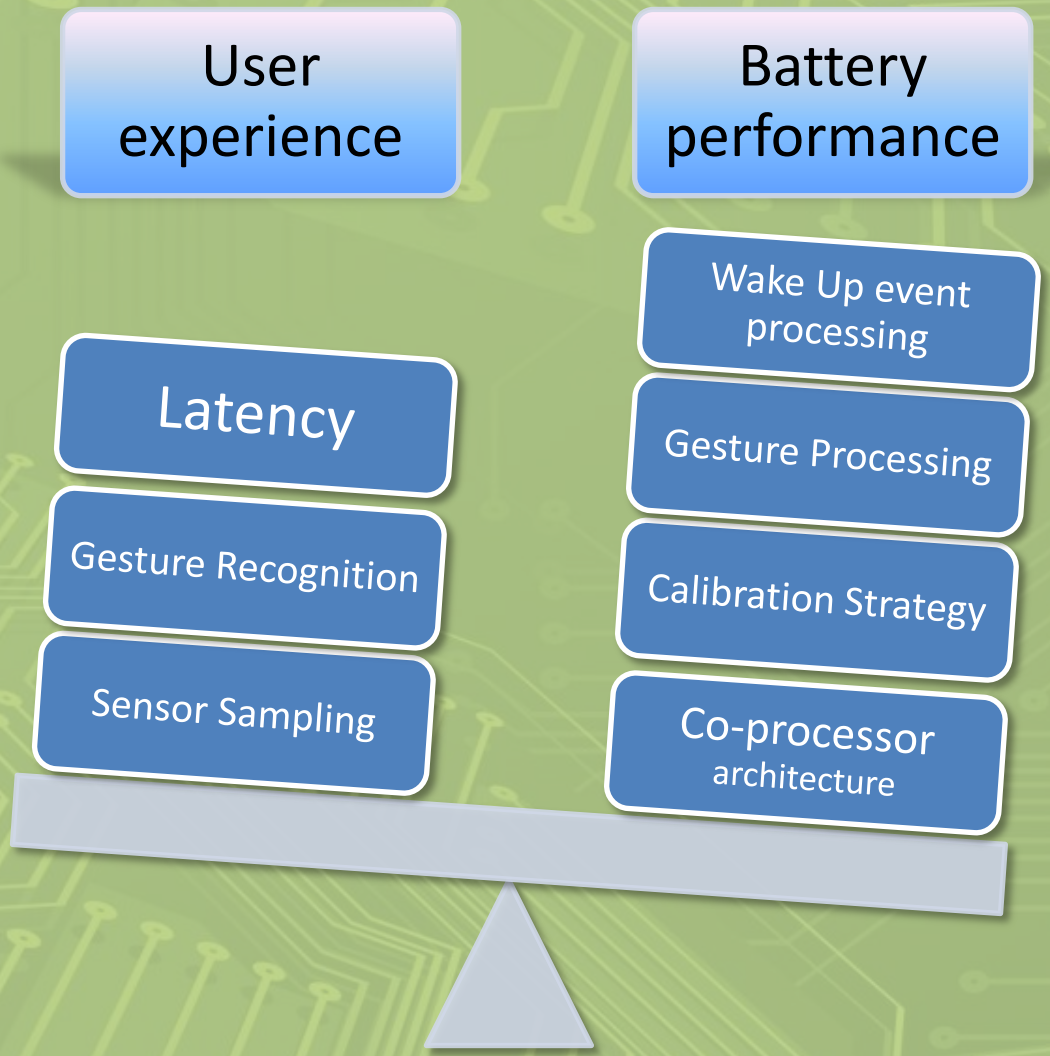
# Designing An Android Sensor Subsystem

**Pitfalls and Considerations**

Jen Costillo

jen@rebelbot.com

# Simple Choices



# Established or Innovative Product?

**Established**

- Will I be making another new product in 6 months?
- Is the reference design considered good enough for the application?

**Innovation-Driven**

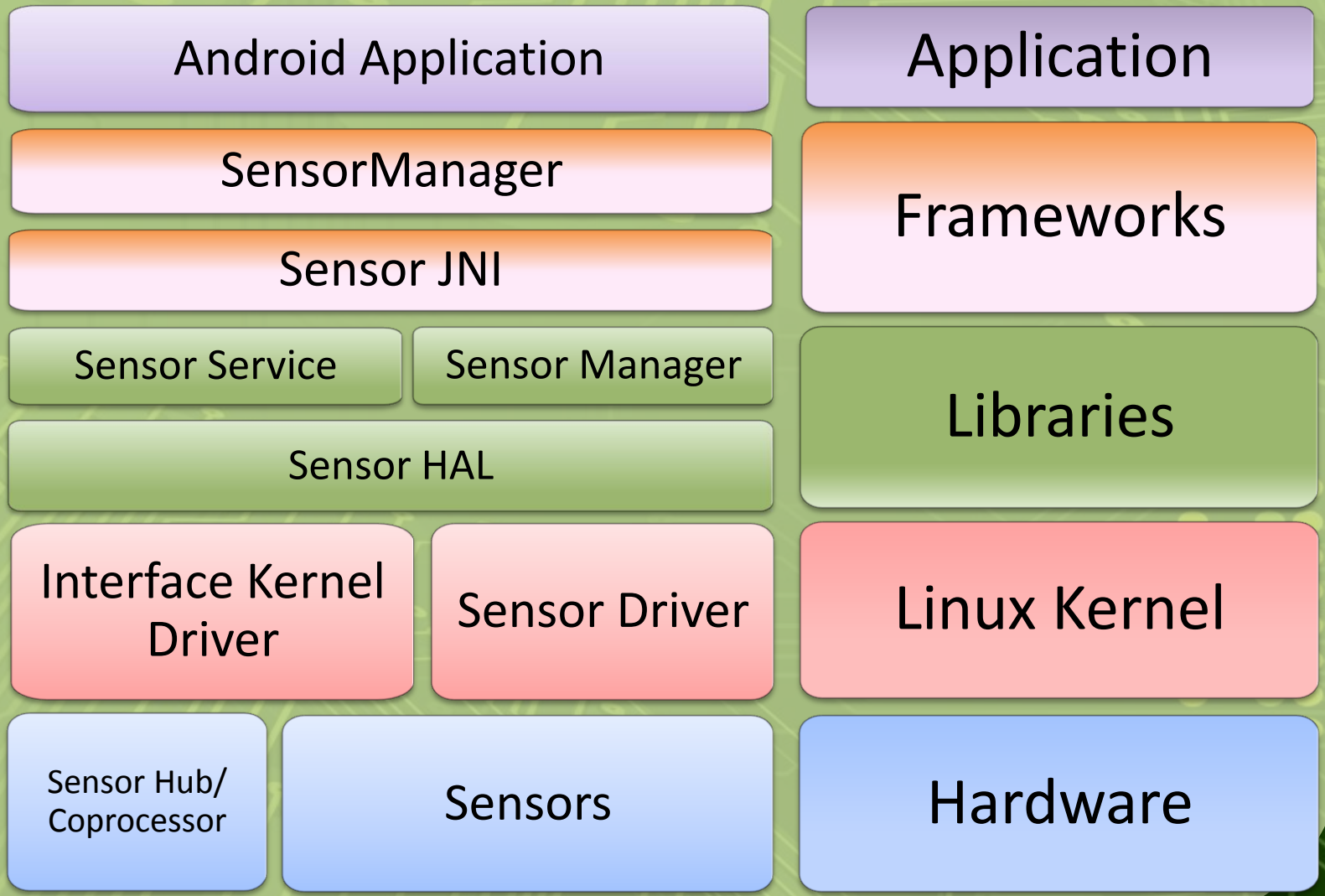
- Do I have new sensors types?
- Are features more important than release date?
- Are money and resources no problem?

# Forsaking Reference Designs

# Going On Your Own

- If you make your own,
  - You're on your own
  - Integration pains
  - Test time ↑
  - Gesture testing becomes a challenge
  - Calibration blues
  - Larger mechanical footprint
- But…
  - power ↓
  - Control code size
  - Control mechanical footprint
  - In-house expertise

# Android Universe



# HARDWARE

# Hardware Architecture

# Sensor Selection

- Limited types
- New type
- Latency
- Power consumption

# Sensor Sampling Rate

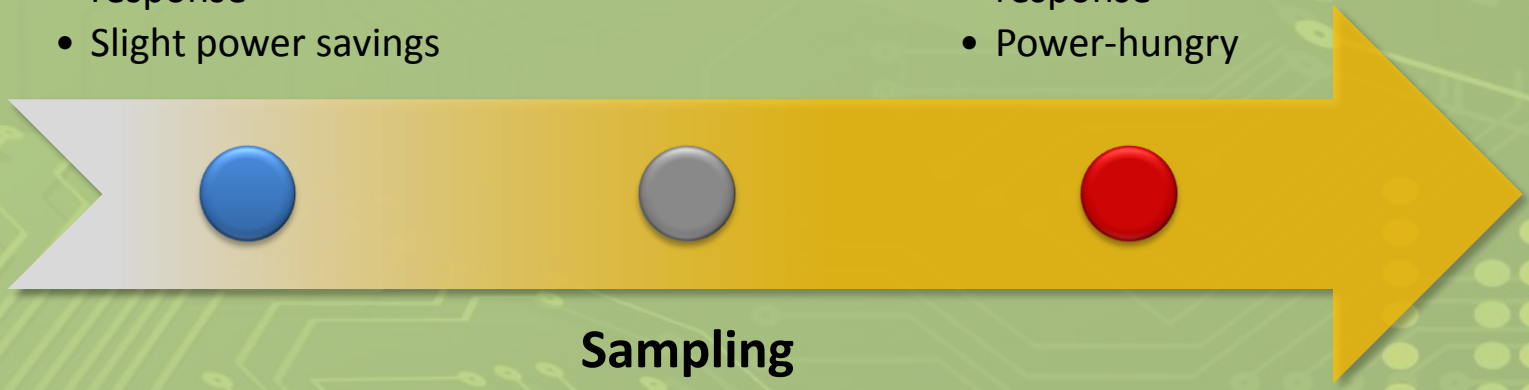
# Sampling Rates: ★ The 3 Rates

**Under-sampling**

- Inaccurate, sluggish response
- Slight power savings

**Over-sampling**

- Accurate, smooth response
- Power-hungry



**Sampling Rate**

# Polling versus Interrupt

**Pros:**

- Simplicity
- Throttle data throughput

**Pros:**

- Low power Sleep Mode
- Use fewer timers

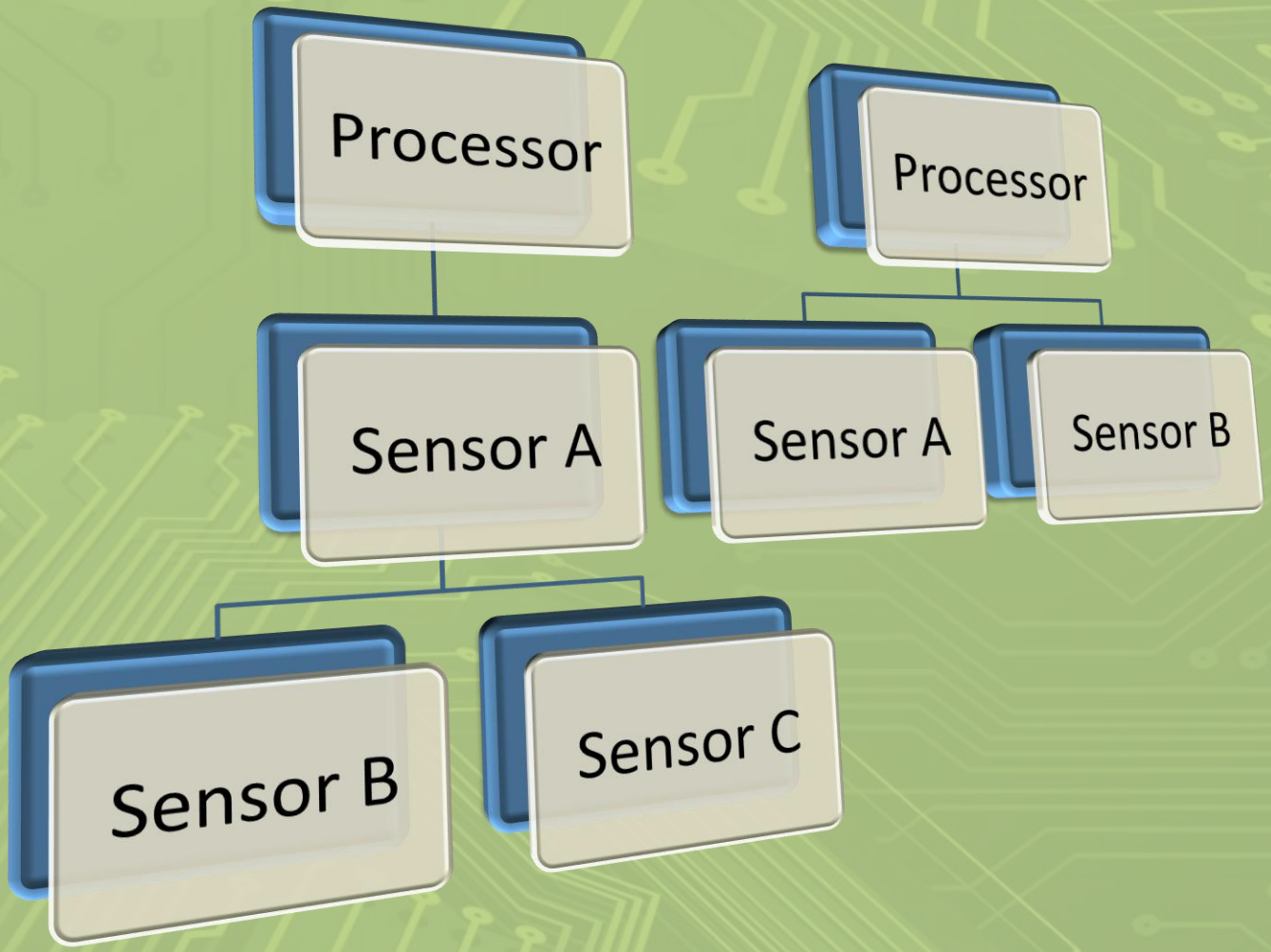
**Cons:**

- Less sleep
- Latency ↑
- Data loss

**Cons:**

- Complex program structure

# Interfaces


Processor
Processor
Sensor A
Sensor A
Sensor B
Sensor B
Sensor C

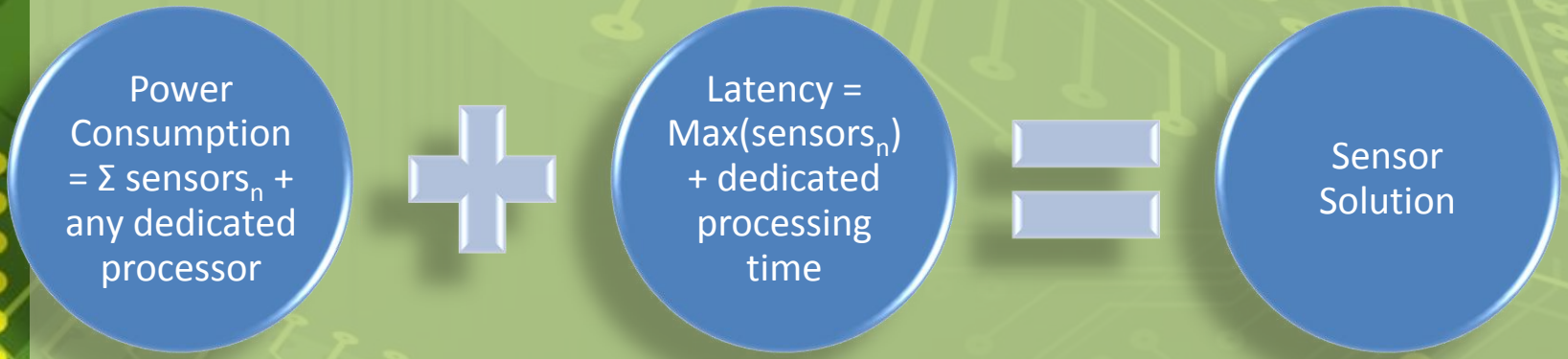
# Wake up events and power considerations ★

| Application Processor only | Internal Coprocessor | External Processor |
|---|---|---|
|  |  |  |
| Reference supported<br>Most power hungry | Reference supported<br>Most work done for you | More processor selection<br>More outcome control<br>Most customized<br>Footprint impact |

# Sensor Subsystem/Hub

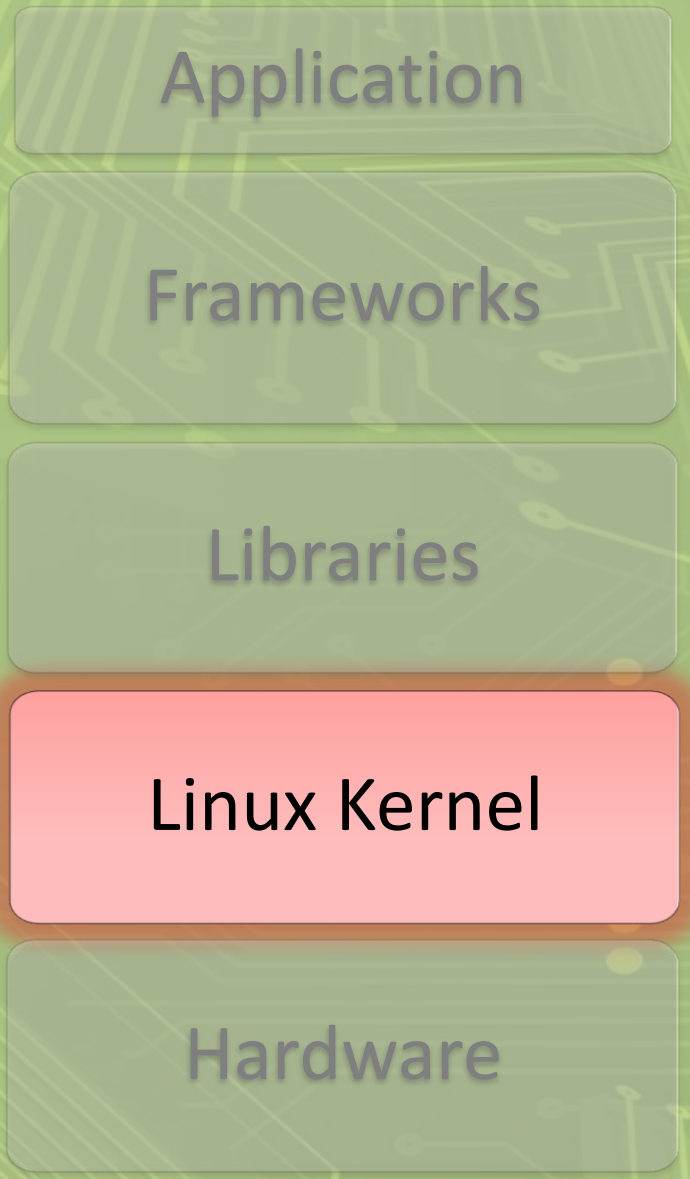
- Separate processor or part of the Application processor
- How to evaluate?
  - *Latency*
  - *Power consumption*
  - *Low power modes*

# Hardware Summary

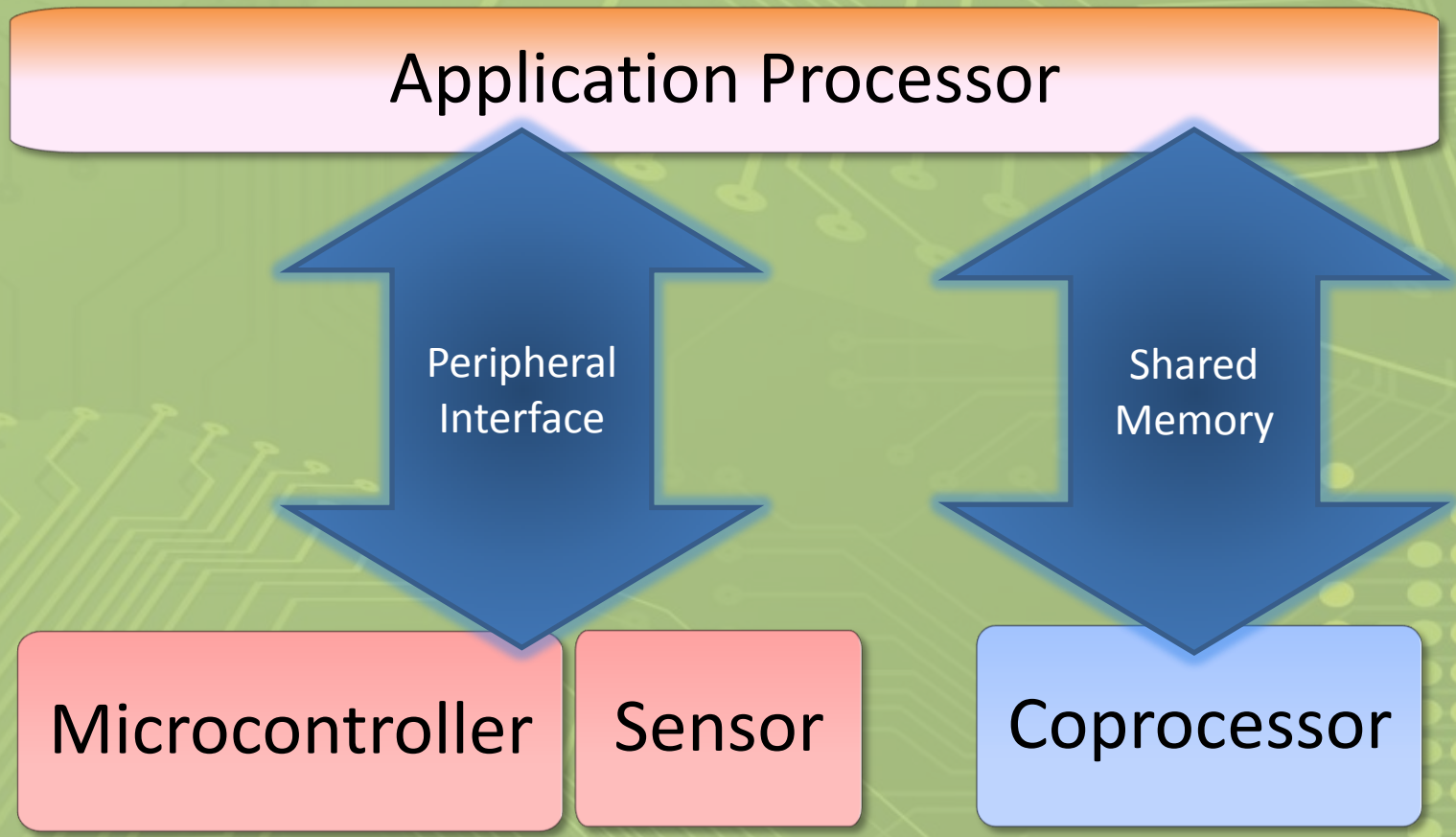



- Use tie-breaker criteria

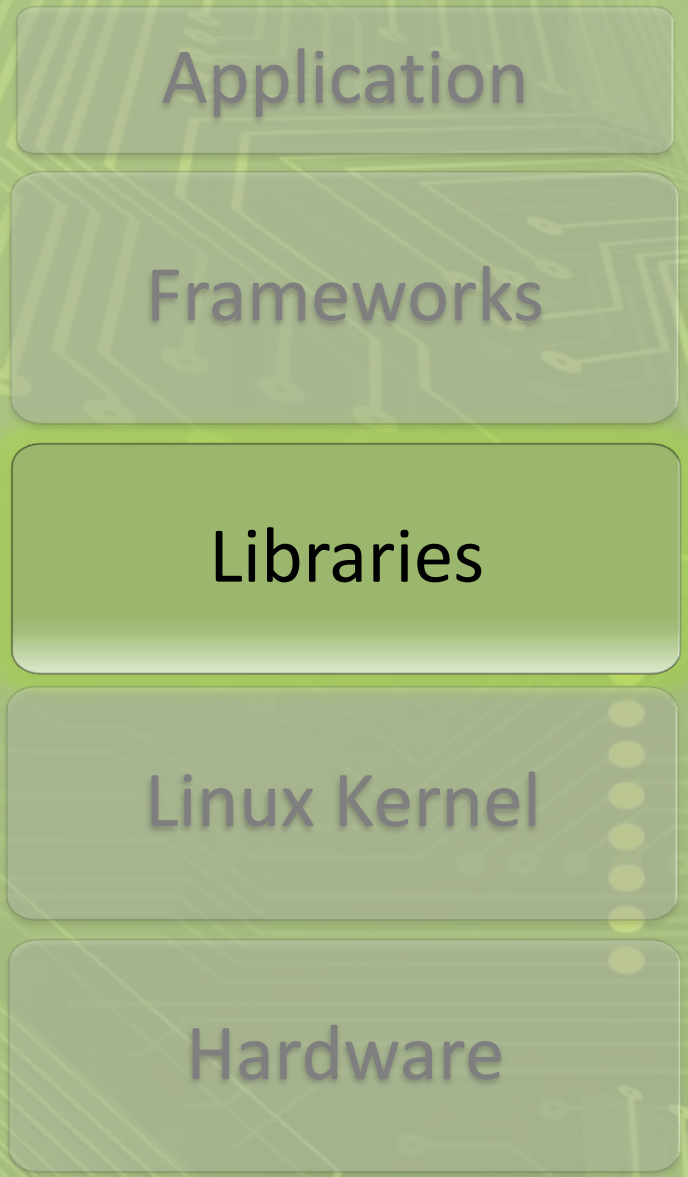
# KERNEL



# Kernel Driver



# LIBRARIES AND SERVICES

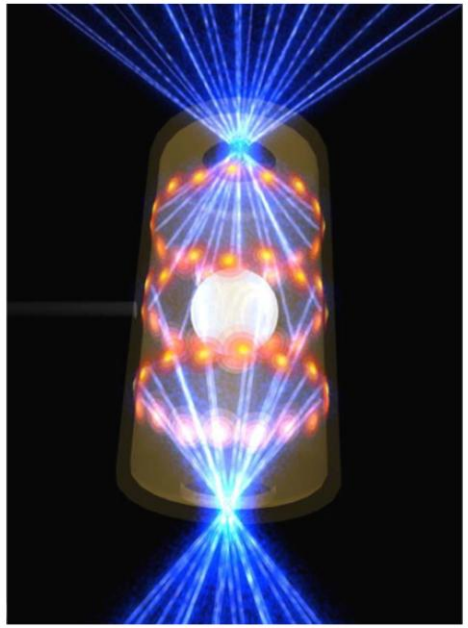


# Sensor HAL and Services

- **HAL**
  device/<vendor>/<board name>/libsensors
- **Service**
  frameworks/base/services/sensorservice
- **Manager**
  frameworks/base/libs/gui

# Sensor Fusion

**Sensor fusion** is the combining of sensory data or data derived from sensory data from disparate sources such that the resulting information is in some sense *better* than would be possible when these sources were used individually. The term



Libraries

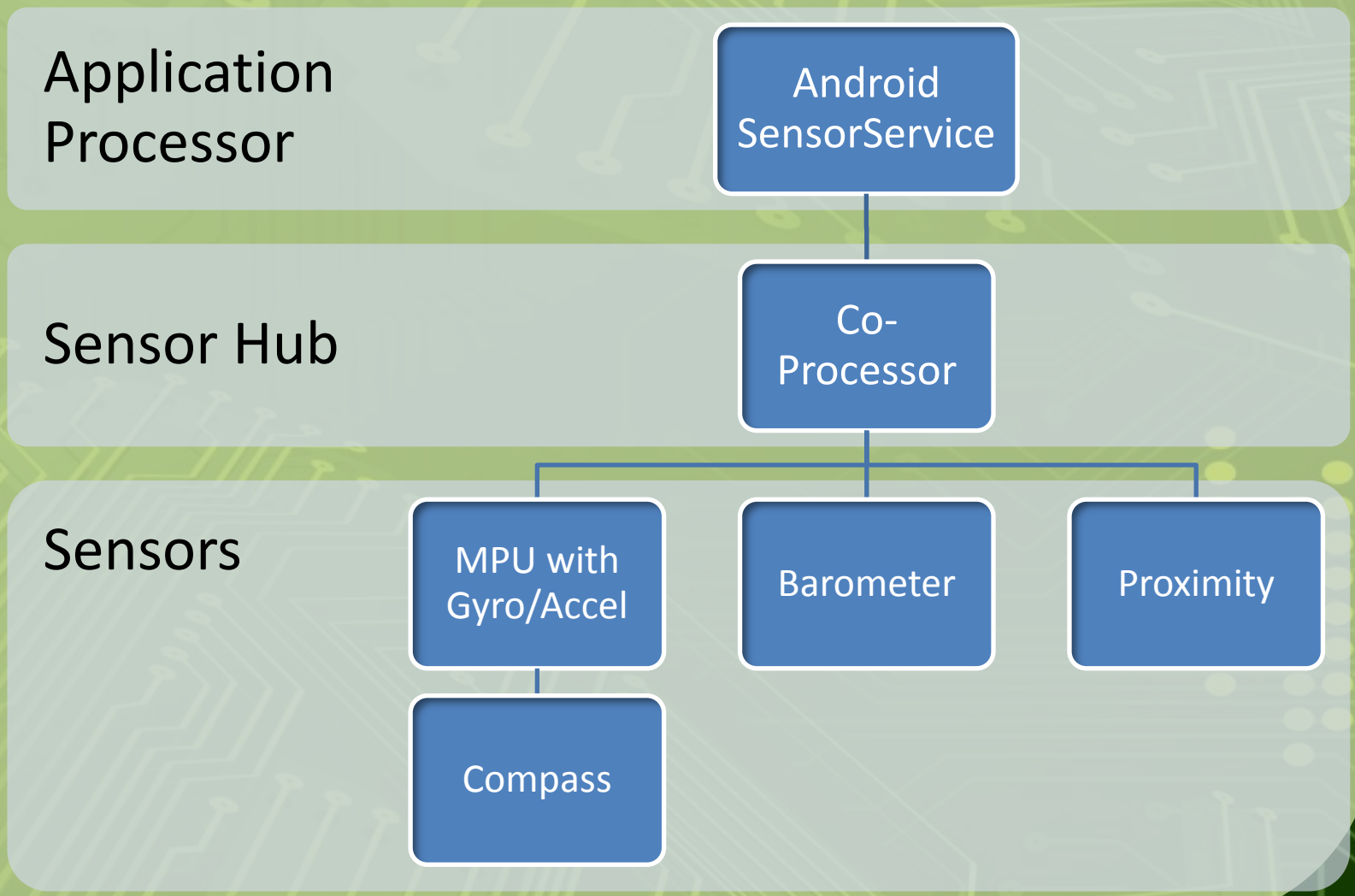
Linux Kernel

Sensor Hub

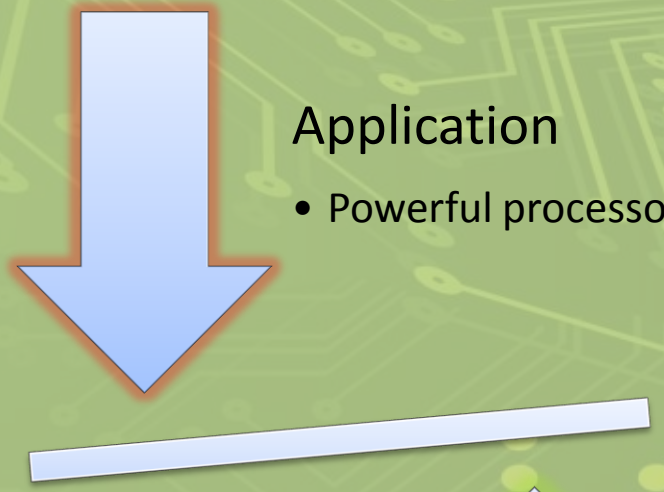
Sensors

http://en.wikipedia.org/wiki/Sensor_fusion
https://www.llnl.gov/news/newsreleases/2010/NR-10-01-06.html

# Gesture Detection Algorithm



# Gesture Detection Comparison

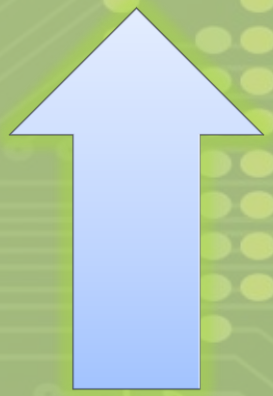



Application

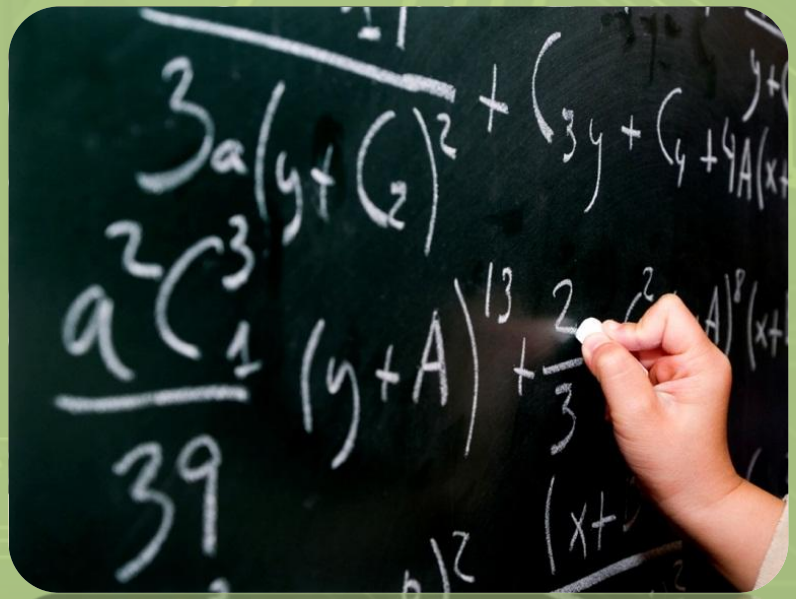
- Powerful processor

Sensor hub

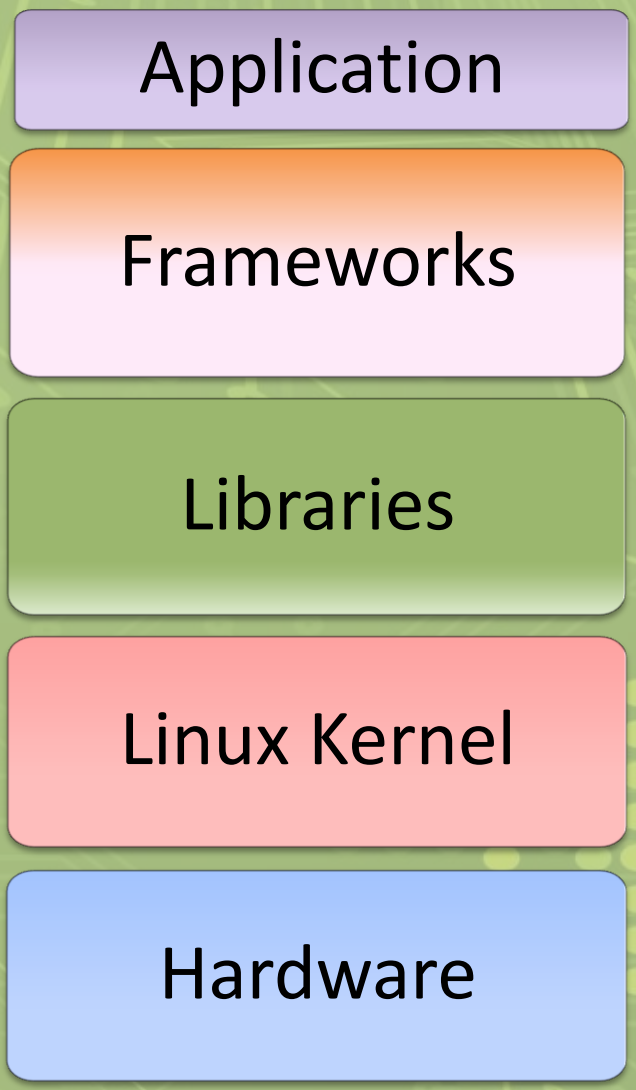
- Off-load to cheaper power
- Wake up Event Handling



# Calibration



# OTHER CONSIDERATIONS

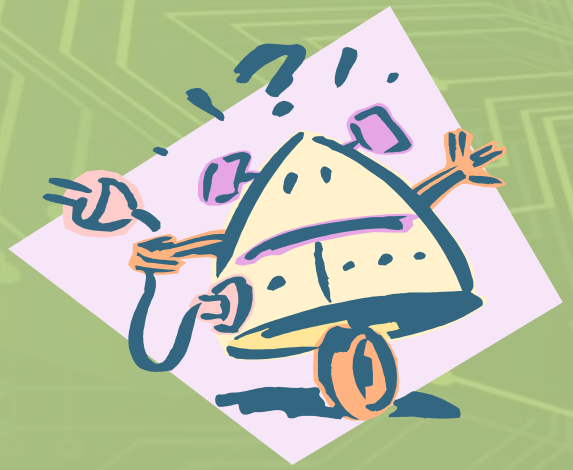


# Testing Methodologies

- Creating tools
- Checkpoints at all levels
- Ensure the application processor can see each of your sensors
- Compatibility Test Suite (CTS) at application level - **/cts/tests/tests/hardware… SensorTest.java**
- Test services - **/frameworks/base/services/sensorservice/tests**
- Manufacturing tests

# QUESTIONS?

JEN@REBELBOT.COM



## Additional resources

http://processors.wiki.ti.com/index.php/Android_Sensor_PortingGuide

http://www.kandroid.org/online-pdk/guide/sensors.html