Implementing Software on Resource-Constrained Mobile Sensors
Experience with Impala and ZebraNet

T. Liu, C. Sadler, P. Zhang, and M. Martonosi, MobiSys '04

Presented by Fabián E. Bustamante (based on the authors' presentation)

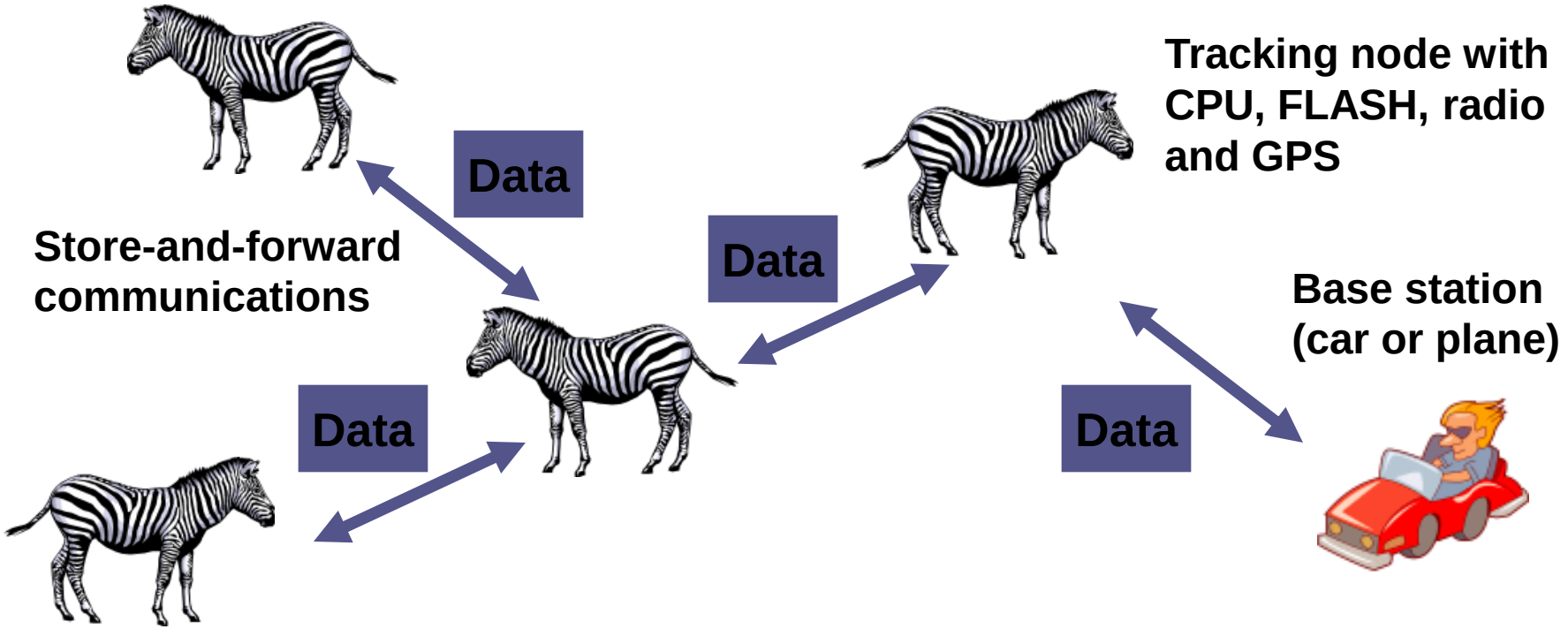
Tracking wildlife

- Focus of biology and biocomplexity research
  - Gather data for different species on interaction and mutual influence
    - E.g. How does human development impact indigenous species?
- Current tracking technology
  - Most common VHF transmitter collar with triangulation
    - Researchers drive through/fly over with a receiver antenna to listen for pings and then track animals
    - Infrequent data collection, mostly during the day
  - Small GPS-based trackers
    - Commonly must retrieve collar to get data
    - Few observation points, no recharge
    - Satellite uploads – high energy and low bitrate

ZebraNet

- Collaboration between CS & wildlife biologist researchers
- Biologist goals
  - Detail positioning information (GPS position every 3')
  - Operation over a large area (1,000km^2), for long unattended periods (1 year)
  - Weight bet/ 3-5lbs
  - No fixed base stations, antennas, etc.
  - Delay tolerant
- Implications
  - Delivery of position data should ~100%
  - Weight limits restrict power availability
  - Large areas means store-and-forward
  - Long-term use in the wild means on-the-fly software updates

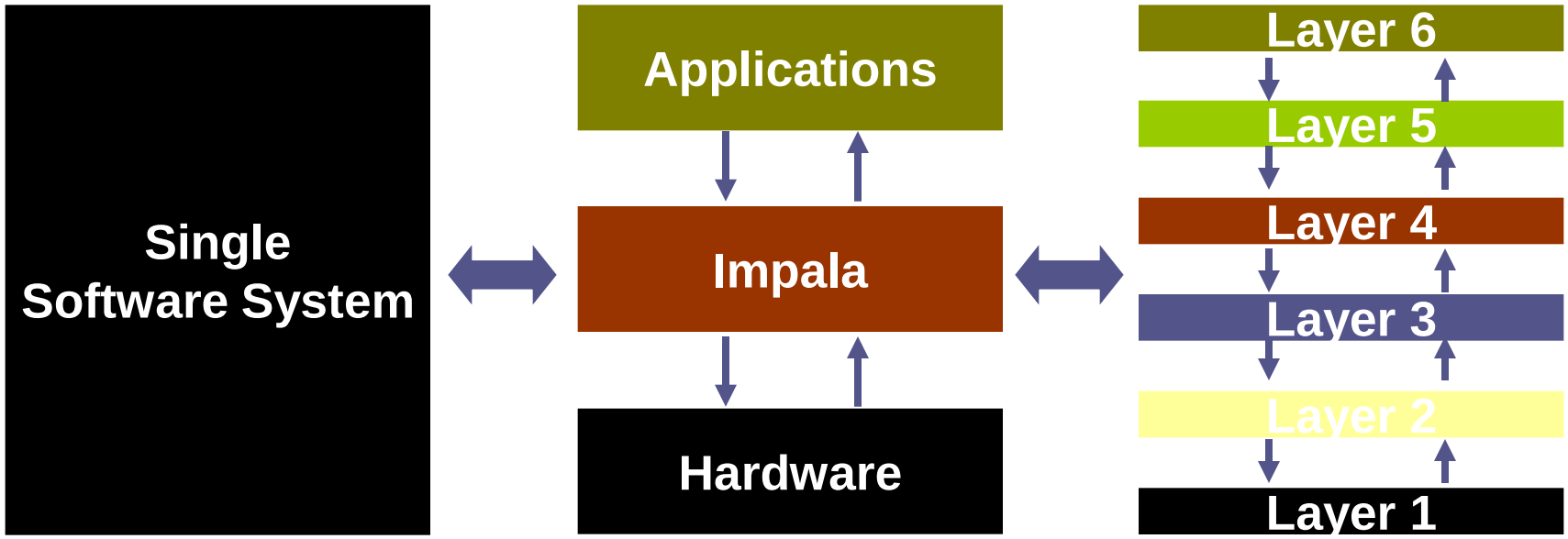
ZebraNet



WSN Attributes	ZebraNet	Other WSNs
Node mobility	High	Static or moderate
Communication range	Miles	Meters
Sensing frequency	Constant	Sporadic
Sensing device power	Hundreds of mW	Tens of mW

Software design rationales

System layering



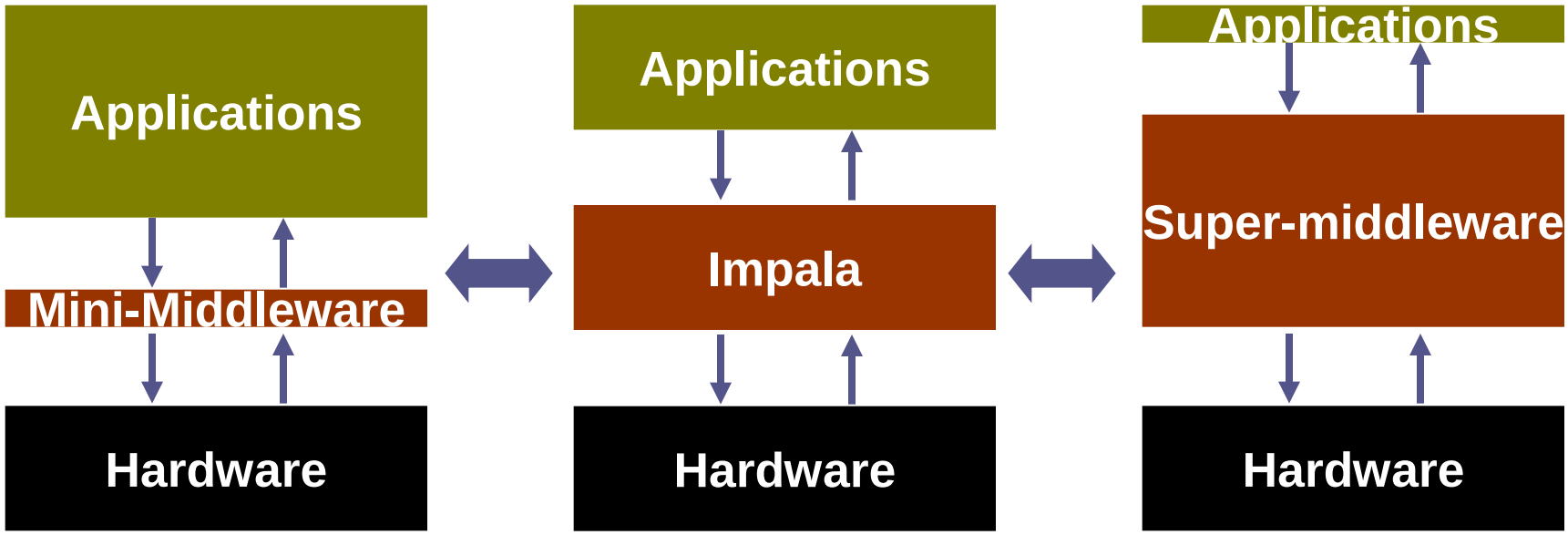
Monolithic approach:
Software modularity

Middle ground

Layered approach:
Layering overhead

Software design rationales

Middleware weight



Limited services: application simplicity

Middle ground

Overloaded services: Middleware overhead

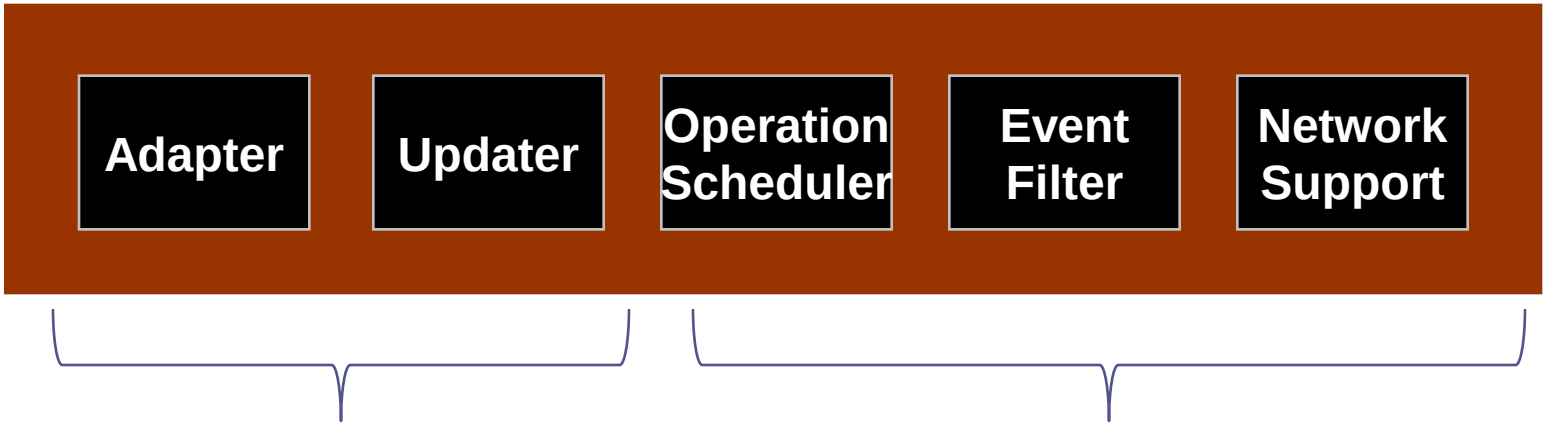
Impala overview

Application modularity, simplicity, adaptivity and repairability

- **Software adaptation for sensor network performance**
- **Software update for sensor network deployment**
- **Operation scheduling for regular operations**
- **Event handling for irregular events**
- **Network support for sensor network communications**

Impala overview

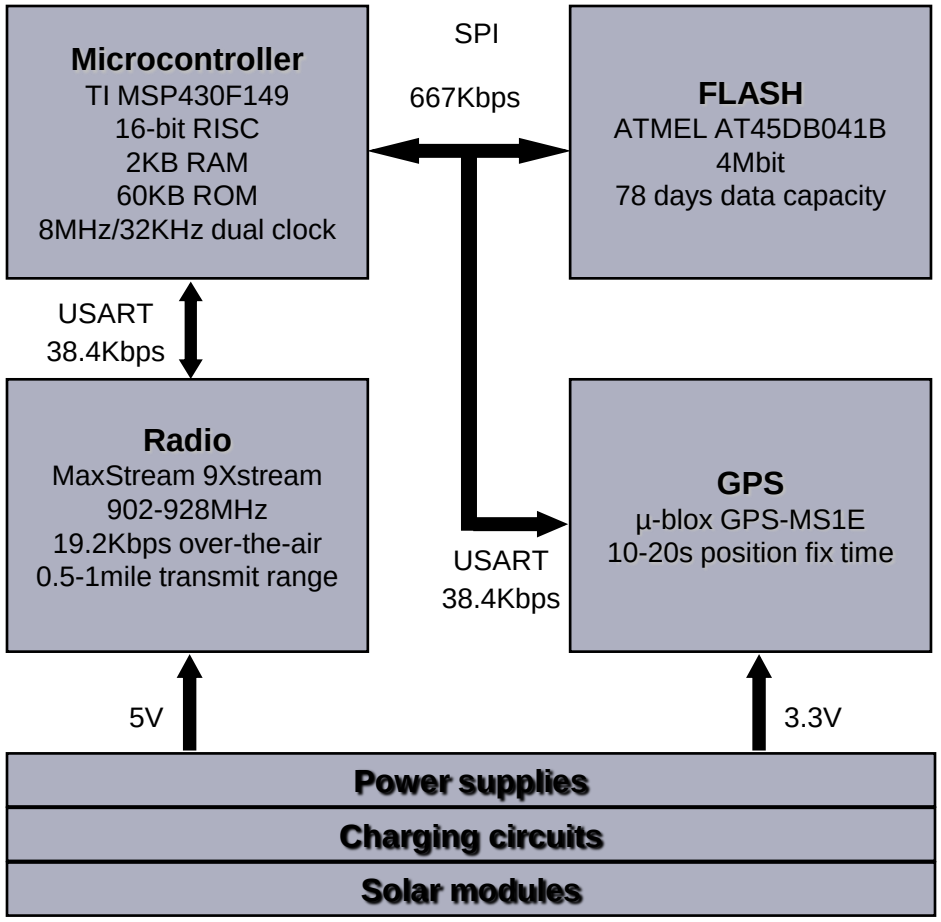
Application modularity, simplicity, adaptivity and repairability



Prototyped on iPAQs

This Paper: on ZebraNet node

Hardware design and constraints

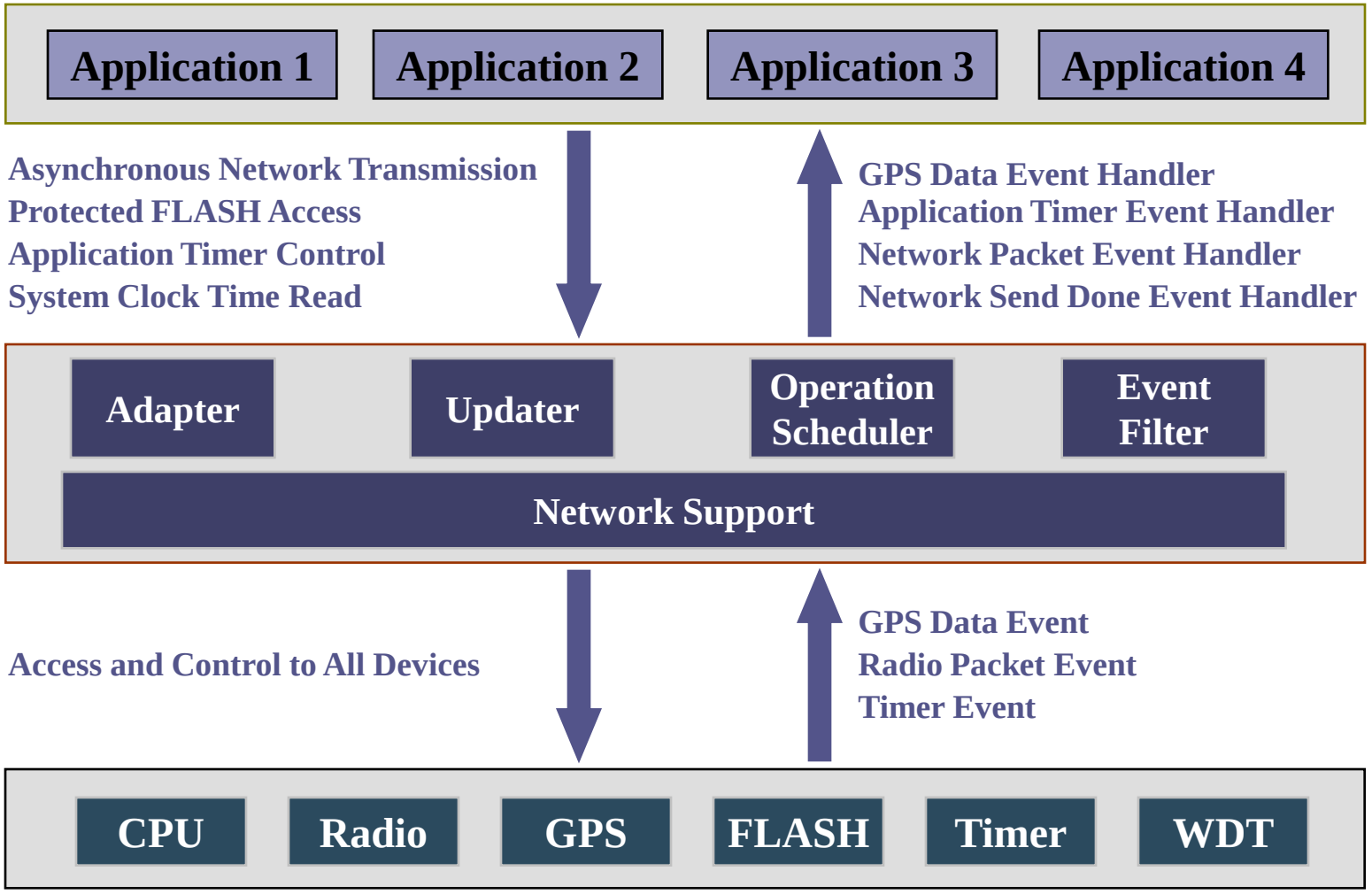


Power Measurement (4.0V applied)

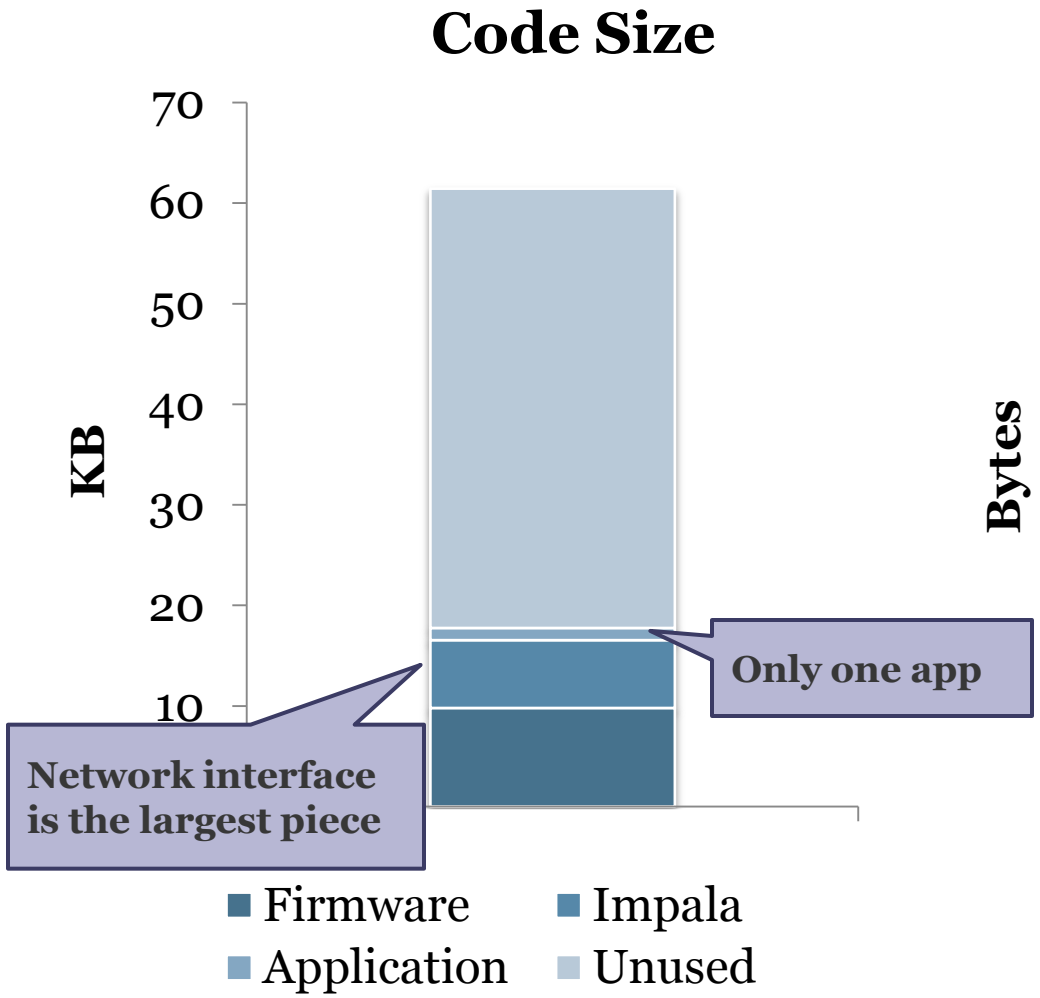
System Mode	Power
CPU at 32KHz	9.6 mW
CPU at 8MHz	19.32 mW
8MHz w/ GPS	568 mW
8MHz w/ radio transmit	780 mW
8MHz w/ radio receive	312.4 mW

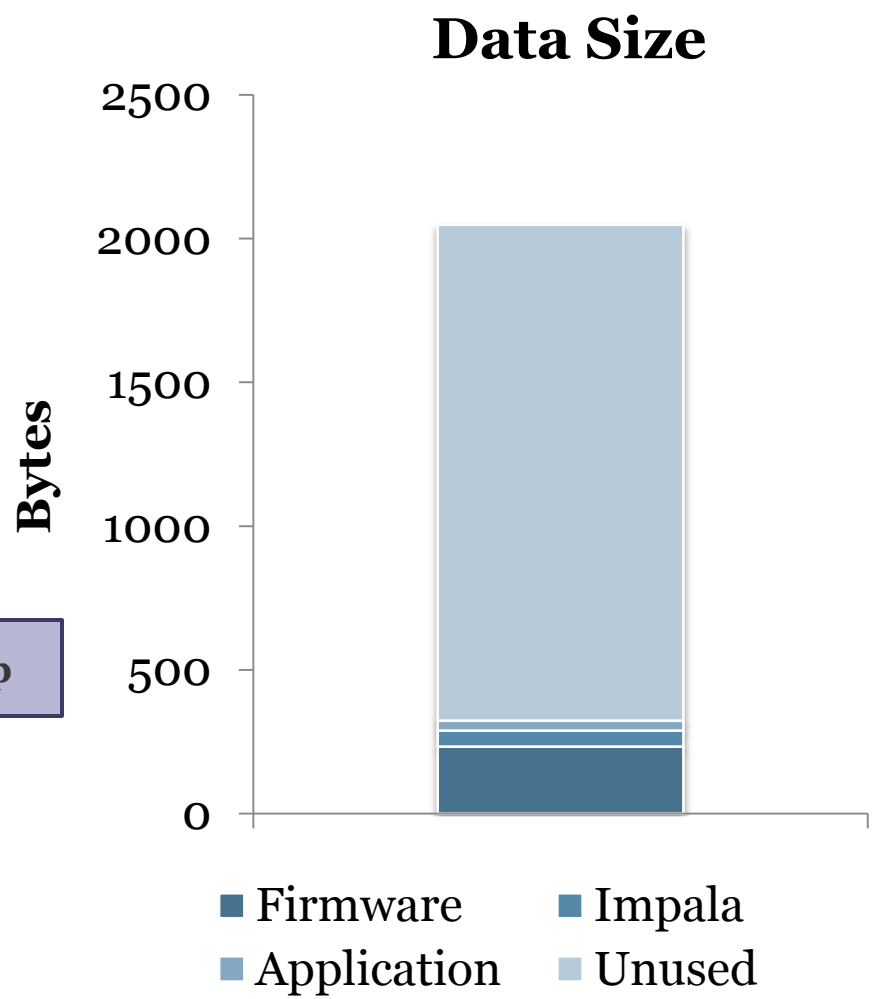
- Memory constraint: 2KB for data, 60KB for program
- Energy constraint: 3 days, 50hrs to recharge
- Device access constraint: GPS and radio interference, e.g.
- Radio packet size constraint: 64B
- GPS sensing time constraint
- FLASH storage constraint

Layers & I/F for complex hardware



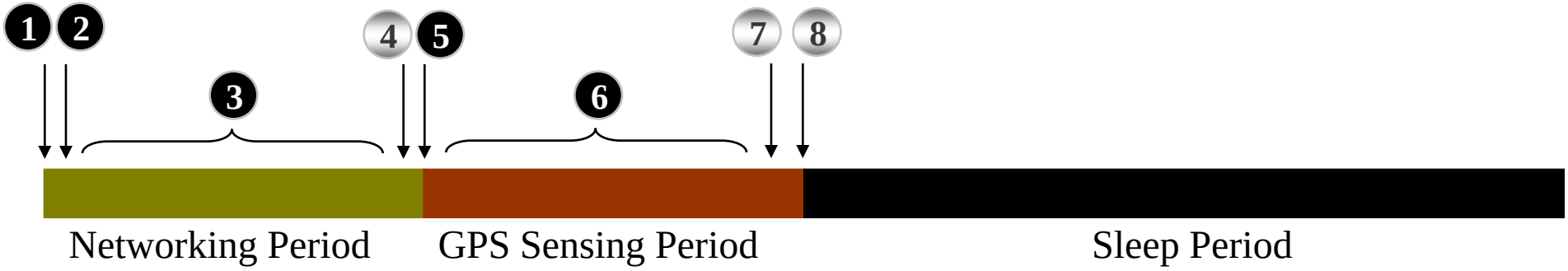
Memory footprint of Impala layers





Regular operation scheduling

- GPS-aided operation synchronization
- Prohibited simultaneous device operations
- Non-trivial radio wake-up time
- Potentially long GPS sensing time (10-40 sec for fixing)
  - Split transaction
- Stringent energy budget



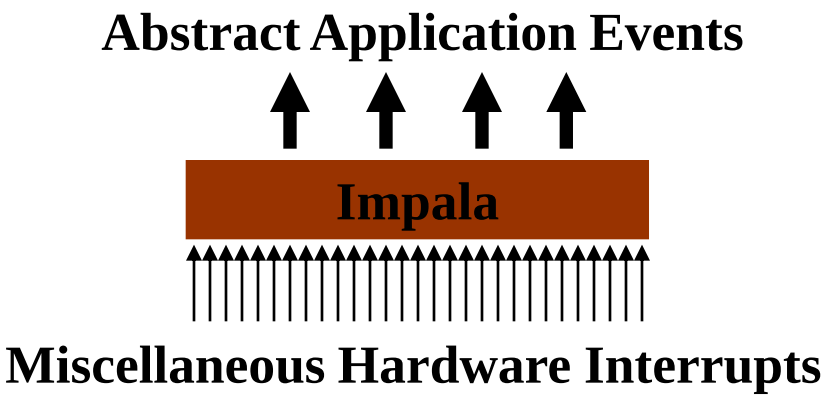
1. CPU wake up/Radio, FLASH power on
2. Radio transmitting / receiving start
3. Network communication time
4. Radio power off/FLASH power off
5. GPS power on
6. GPS sensing time
7. GPS power off / FLASH power on
8. CPU sleep / FLASH power off

Operation scheduling overhead

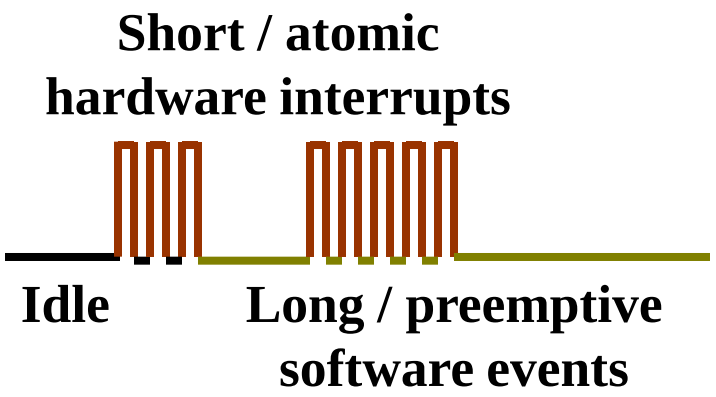
Scheduling Type	Impala Activity	Time (cycles)
CPU Scheduling	To put CPU on the full-speed clock	3127
	To put CPU on the slow clock	38
Radio and FLASH Scheduling	To set up the first transmission time and turn on radio and FLASH	50 ms
	To set up the next transmission time	260
	To set up the network cleanup time	265
	To clean up incomplete network messages, power off radio and FLASH, and set up the next networking period	11 ms
GPS Scheduling	To initiate GPS sensing and set up its finish time	1247
	To format GPS data, power off GPS, signal an GPS data event, and set up the next GPS sensing period	2550

Handling irregular events

Issue One: Event Abstraction

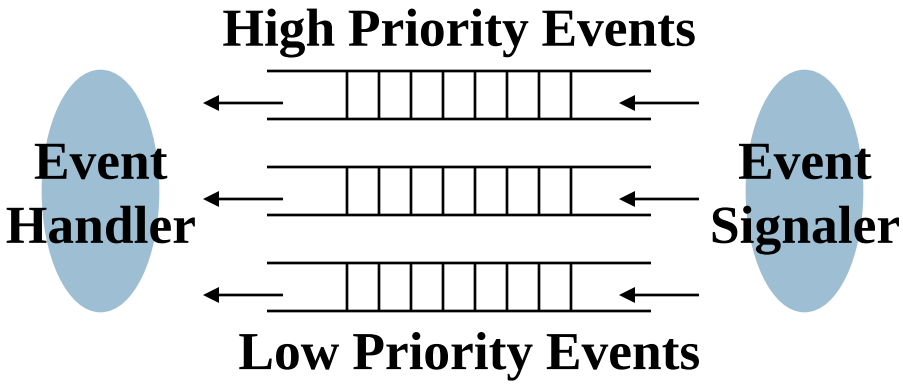


Issue Two: Concurrency



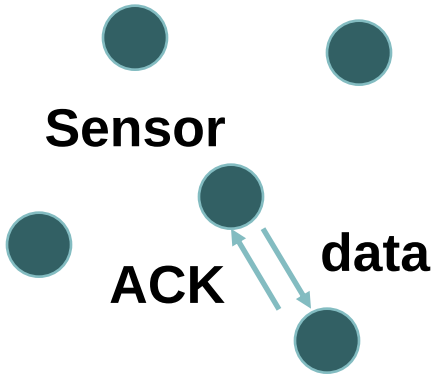
Issue Three: Event Prioritization

A priority ordered set of queues, one for each event type

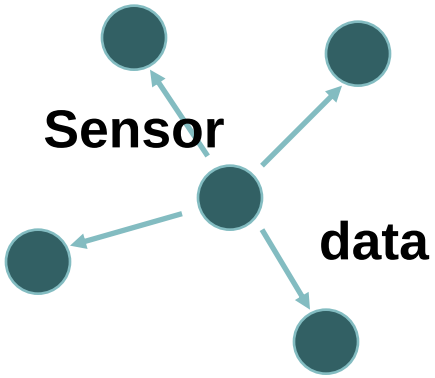


Special communication needs

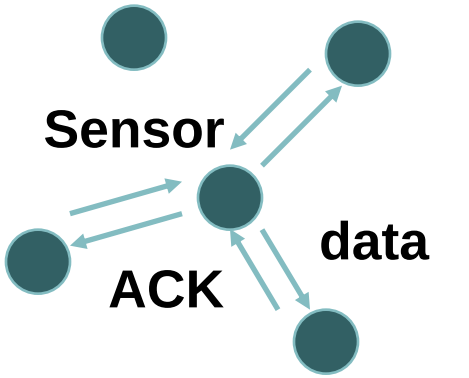
- Peer discovery or data
- Session-based
  - Msg designated by the app to have network transaction semantics
  - 1 to 32KB
  - Header information in != packet
- One, more or unlimited destinations
- Data from FLASH or RAM
- Acknowledgment or not
- Connectionless



Reliable Unicast



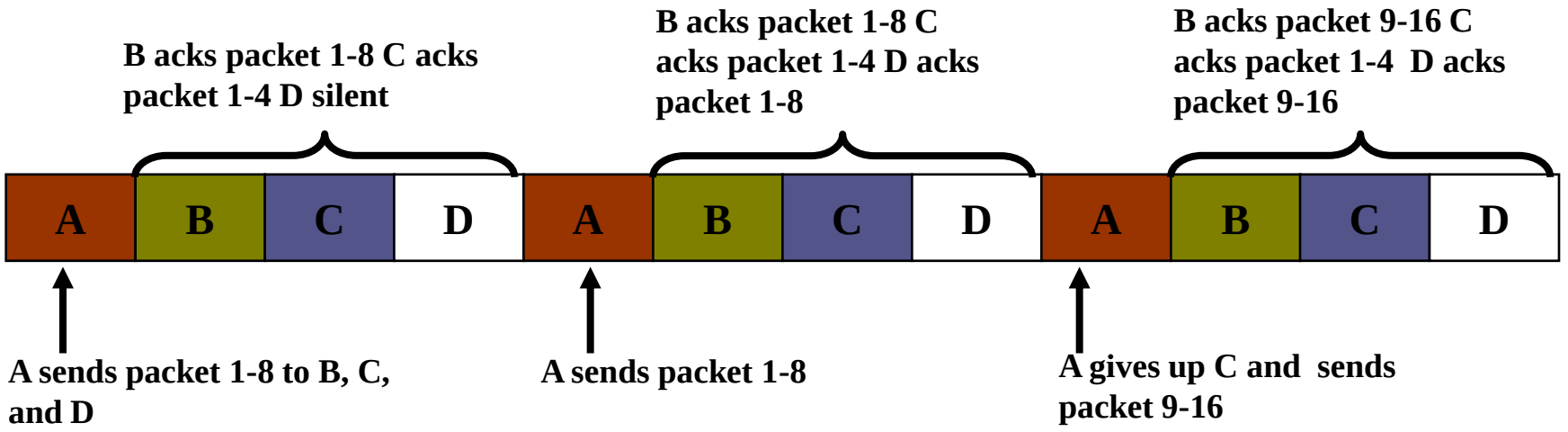
Unreliable Broadcast



Reliable Multicast

Unified MAC & transport control

- MAC: round-robin timeslots (w/ GPS-aided time synchronization)
- Transport control: detect packet loss and retransmit
- Unified: reduce data copy & overhead across layers
  - Replace data buffering with indexing
  - Packet, rather than session buffering, in network reception – complicates apps
- Asynchronous



Conclusions

- Propose and implement Impala middleware model
- Solutions for hardware constraints and application requirements
- Concrete experience with real system and application
- To-do: Refine collar design; switch to lower-energy GPS, merge Impala software update code into final collars