IP *is* the Future of Ubiquitous Sensor Networks

TinyOS USN applications workshop

David E. Culler
University of California, Berkeley
Arch Rock Corp.

The Internet … that we envisioned

THE Question

If Wireless Sensor Networks represent a future of "billions of information devices embedded in the physical world,"

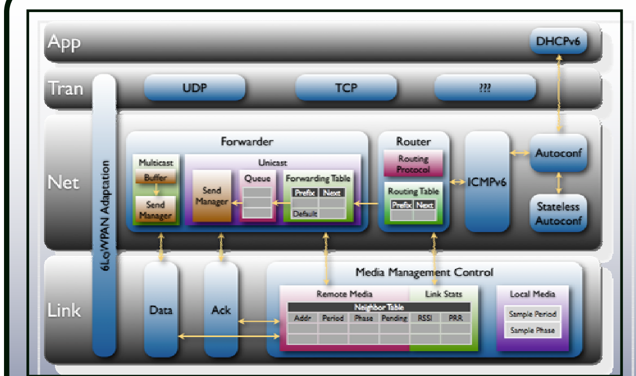
why don't they run ***<u>THE</u>*** standard internetworking protocol?

The Answer

They should

- Substantially advances the state-of-the-art of both.
- Implementing IP requires tackling the general case, not just a specific operational slice
  - Interoperability with all other potential IP network links
  - Potential to name and route to any IP-enabled device within security domain
  - Robust operation despite external factors
    - Coexistence, interference, errant devices, ...
- While meeting the critical embedded wireless requirements
  - High reliability and adaptability
  - Long lifetime on limited energy
  - Manageability of many devices
  - Within highly constrained resources

IP in TinyOS on Motes is a reality today



24038 ROM

3598 RAM
(including runtime)

* Production implementation on TI msp430/cc2420

- Footprint, power, packet size, & bandwidth

	ROM	RAM
CC2420 Driver	3149	272
802.15.4 Encryption	1194	101
Media Access Control	330	9
Media Management Control	1348	20
6LoWPAN + IPv6	2550	0
Checksums	134	0
SLAAC	216	32
DHCPv6 Client	212	3
DHCPv6 Proxy	104	2
ICMPv6	522	0
Unicast Forwarder	1158	451
Multicast Forwarder	352	4
Message Buffers	0	2048
Router	2050	106
UDP	450	6
TCP	1674	50

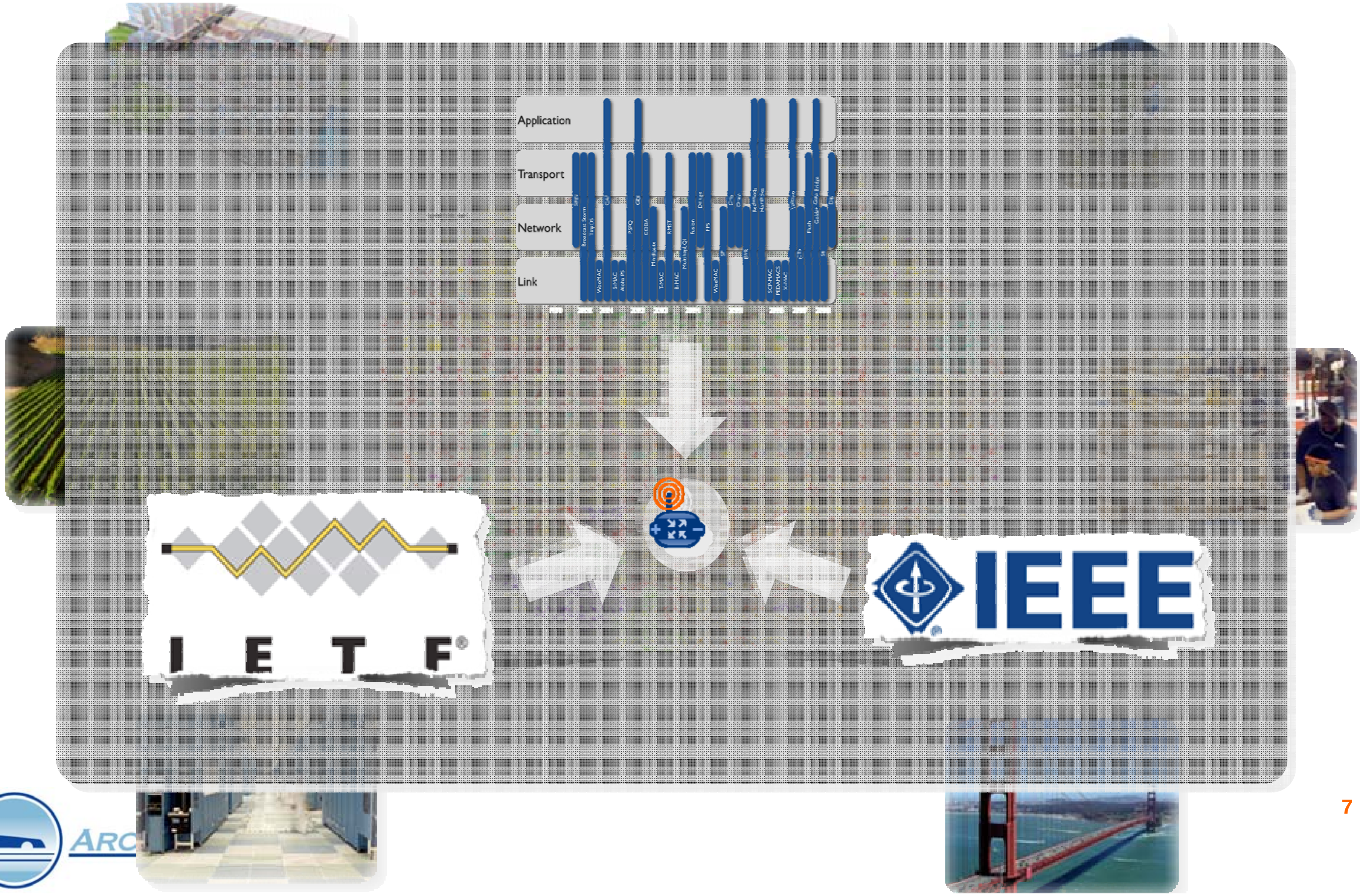
A Decade Ago we could not imagine IP would scale to Tiny WSNs

- “Resource constraints may cause us to give up the layered architecture.”
- “Sheer numbers of devices, and their unattended deployment, will preclude reliance on broadcast communication or the configuration currently needed to deploy and operate networked devices.”
- “There are significant robustness and scalability advantages to designing applications using localized algorithms.”
- “Unlike traditional networks, a sensor node may not need an identity (e.g. address).”
- “It is reasonable to assume that sensor networks can be tailored to the application at hand.”

We were wrong...

ArchRock

Confluence on three fronts

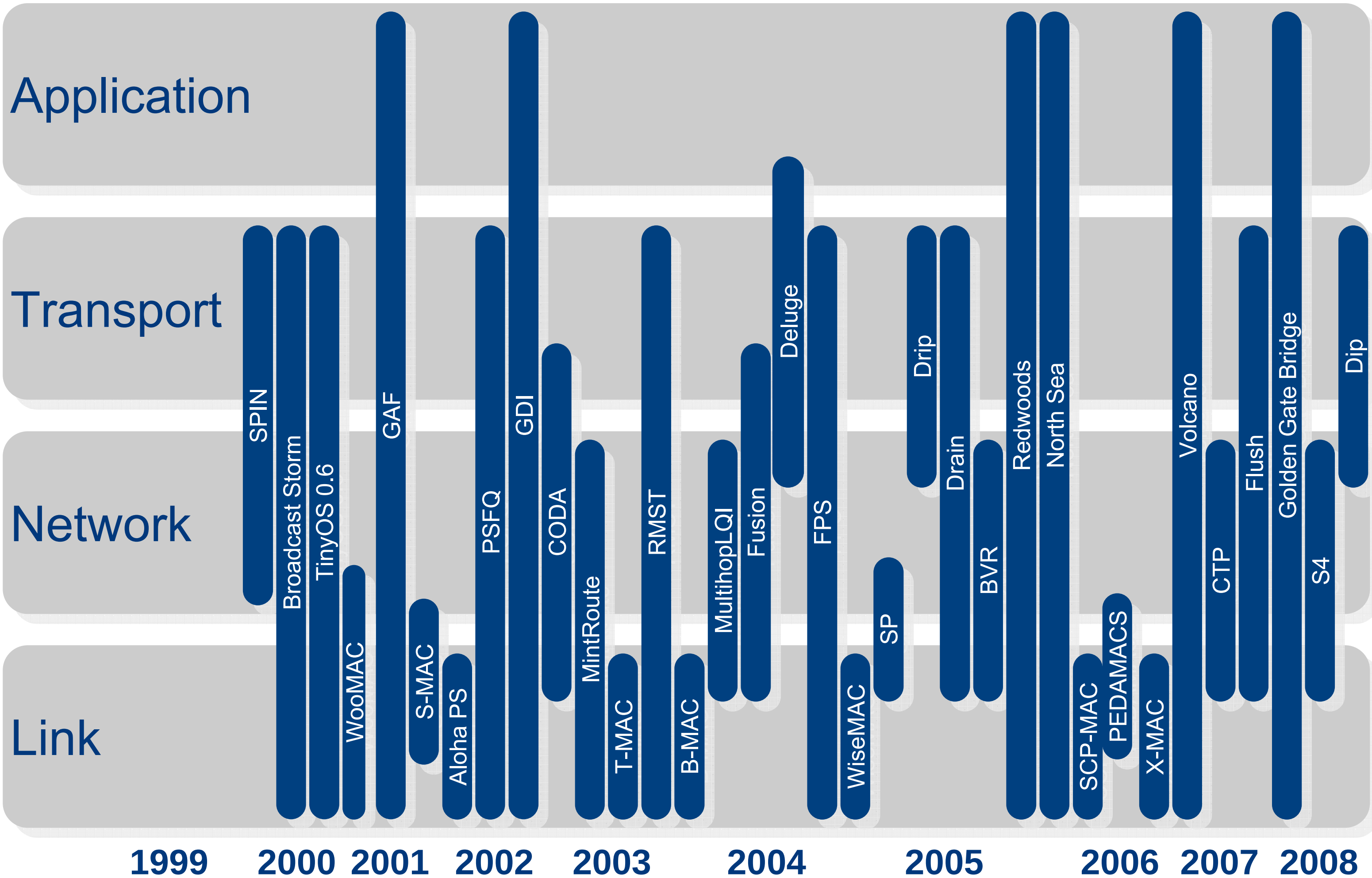


A Low-Power Standard Link

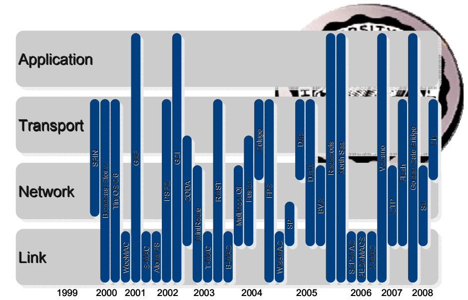
	802.15.4	802.15.1	802.15.3	802.11	802.3
Class	WPAN	WPAN	WPAN	WLAN	LAN
Lifetime (days)	100-1000+	1-7	Powered	0.1-5	Powered
Net Size	65535	7	243	30	1024
BW (kbps)	20-250	720	11,000+	11,000+	100,000+
Range (m)	1-75+	1-10+	10	1-100	185 (wired)
Goals	Low Power, Large Scale, Low Cost	Cable Replacement	Cable Replacement	Throughput	Throughput

- Low Transmit power, Low SNR, modest BW, Little Frames

Decade of Sensor Network Research - without Architecture



Three Key Developments



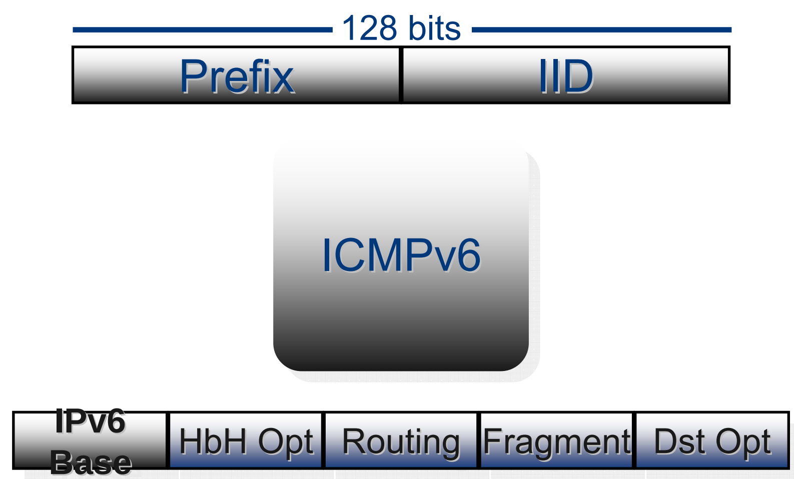
- Idle listening
  - Radio power: transmit ≈ receive ≈ just listening
  - All the energy is consumed by listening for a packet to receive
    - E = P*Time

  => Turn radio on only when there is something to hear
- Reliable routing on Low-Power & Lossy Links
  - Power, Range, Obstructions => multi-hop
  - Always at edge of SNR => loss happens

  => monitoring, retransmission, and local rerouting
- Trickle – don't flood (tx rate < 1/density, and < info change)
  - Connectivity is determined by physical points of interest, not network designer. May have huge number of neighbors, so …
  - never naively respond to a broadcast
  - re-broadcast very very politely

Key IPv6 Contributions

- Large simple address
  - Network ID + Interface ID
  - Plenty of addresses, easy to allocate and manage
- Autoconfiguration and Management
  - ICMPv6
- Integrated bootstrap and discovery
  - Neighbors, routers, DHCP
- Protocol options framework
  - Plan for extensibility
- Simplify for speed
  - MTU discovery with min
- 6-to-4 translation for compatibility

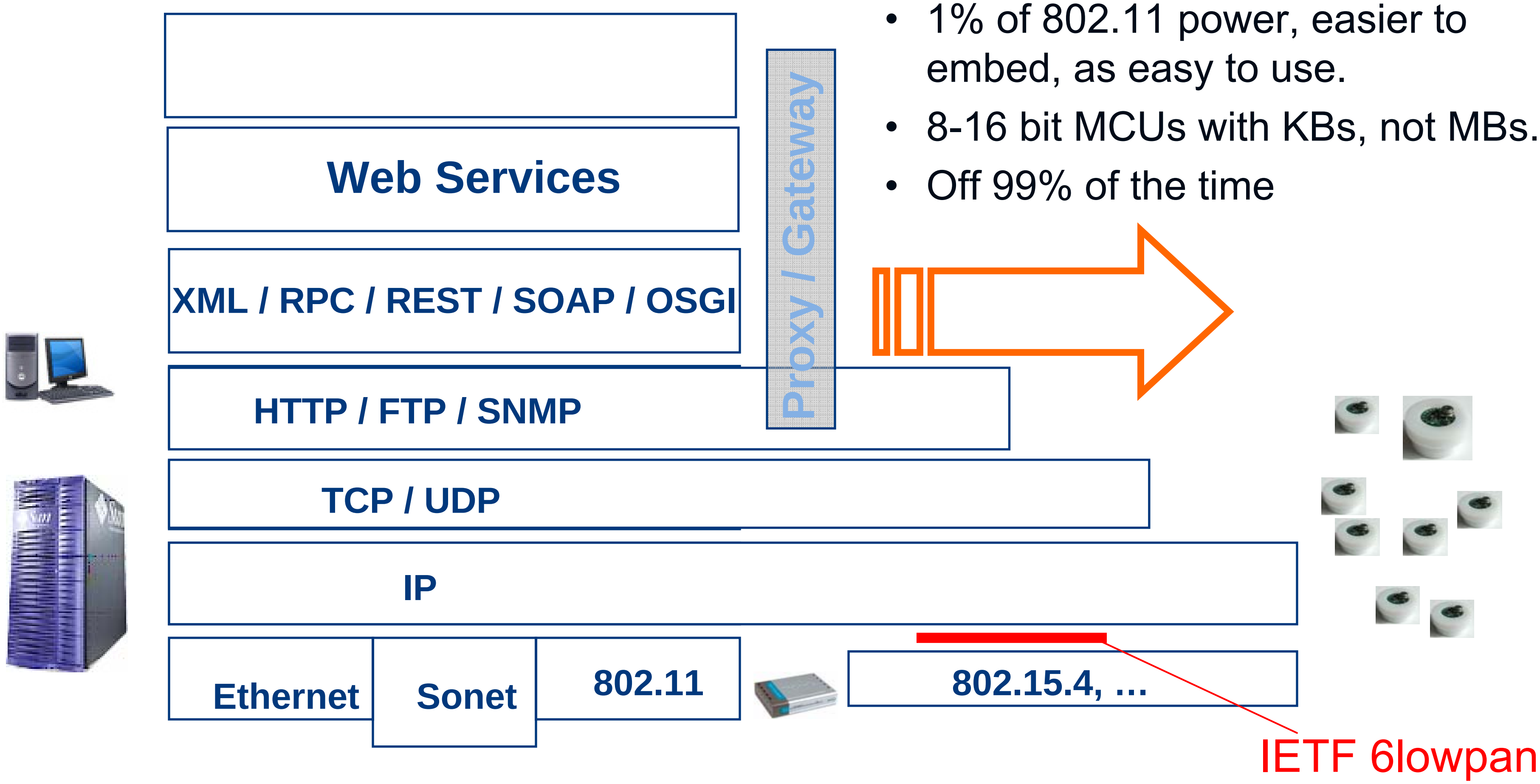


ARCHROCK

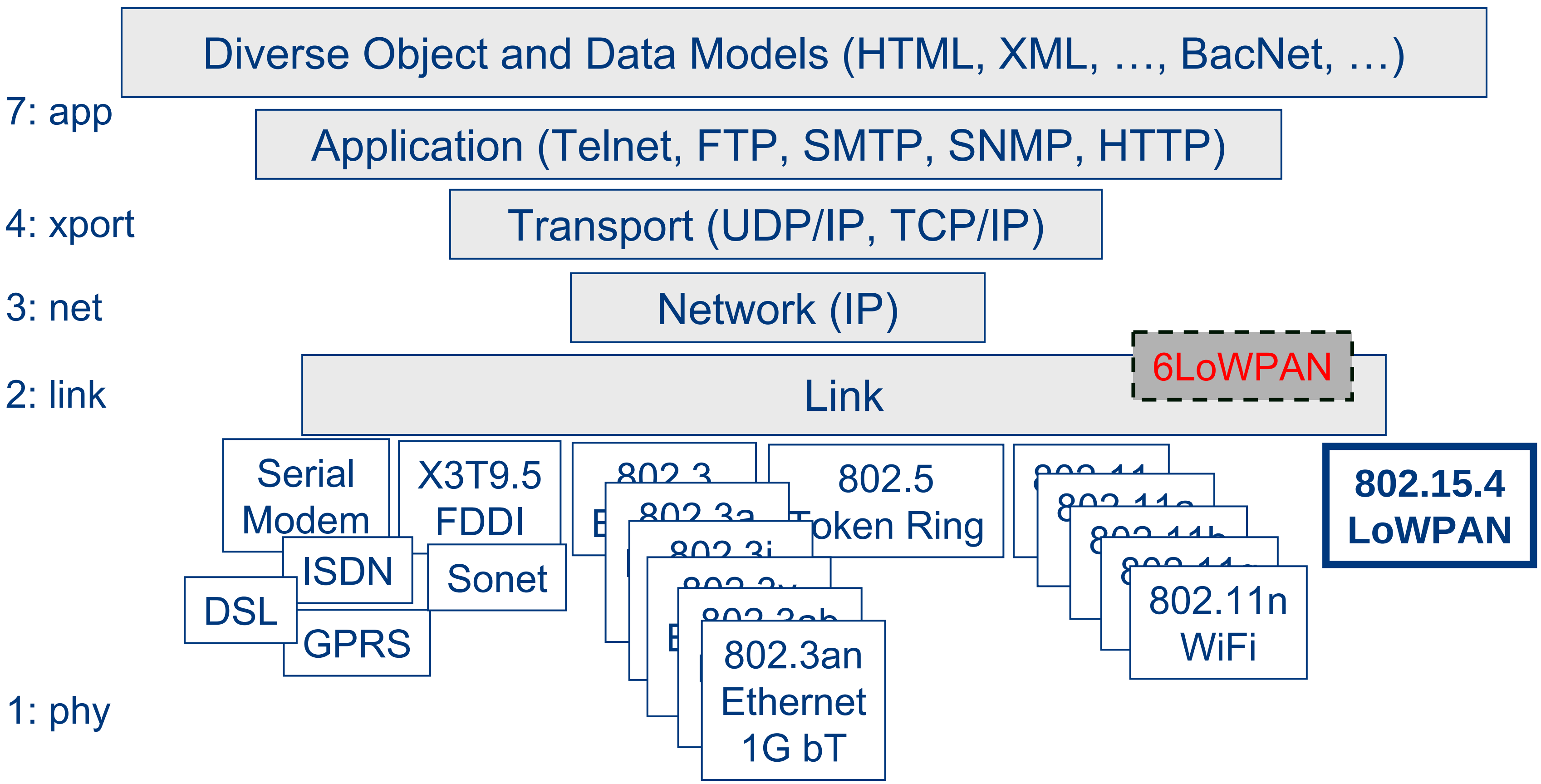
Making sensor nets make sense

LoWPAN – 802.15.4

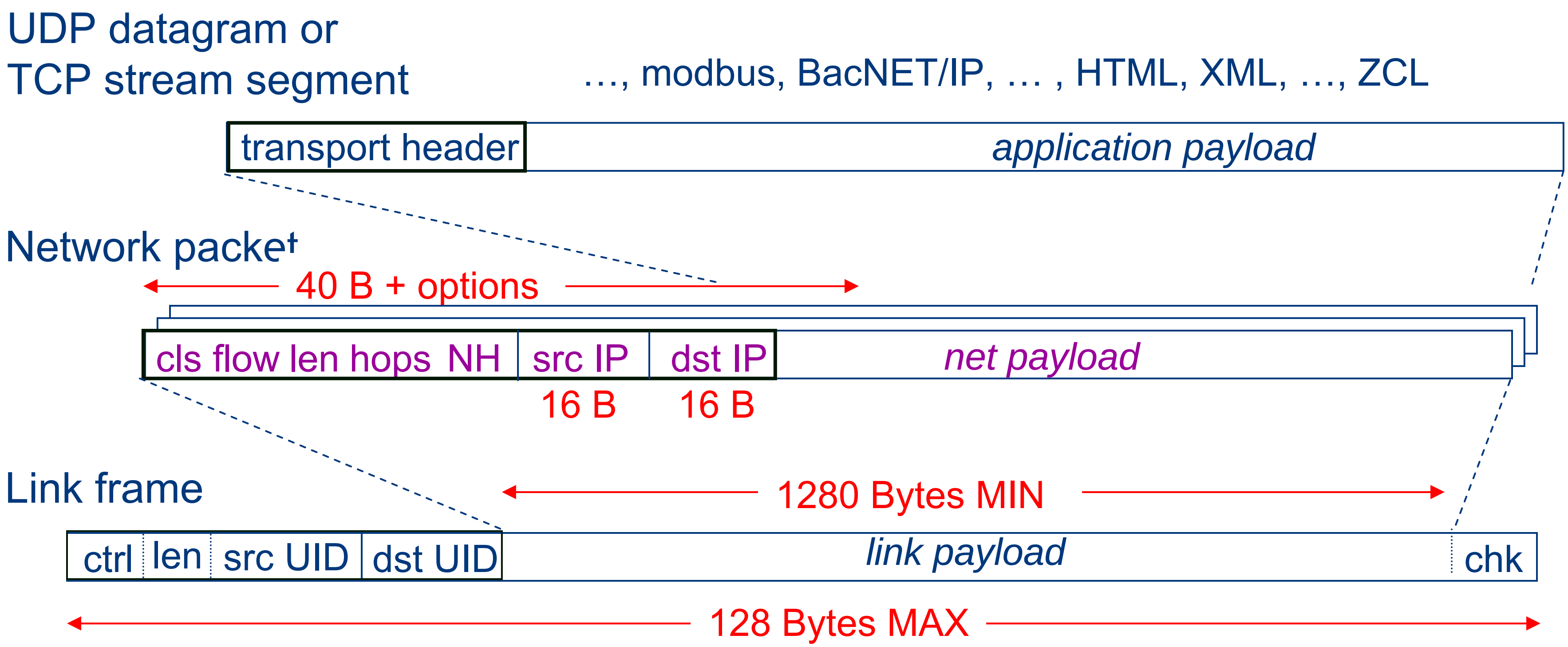
- 1% of 802.11 power, easier to embed, as easy to use.
- 8-16 bit MCUs with KBs, not MBs.
- Off 99% of the time



6LoWPAN adaptation



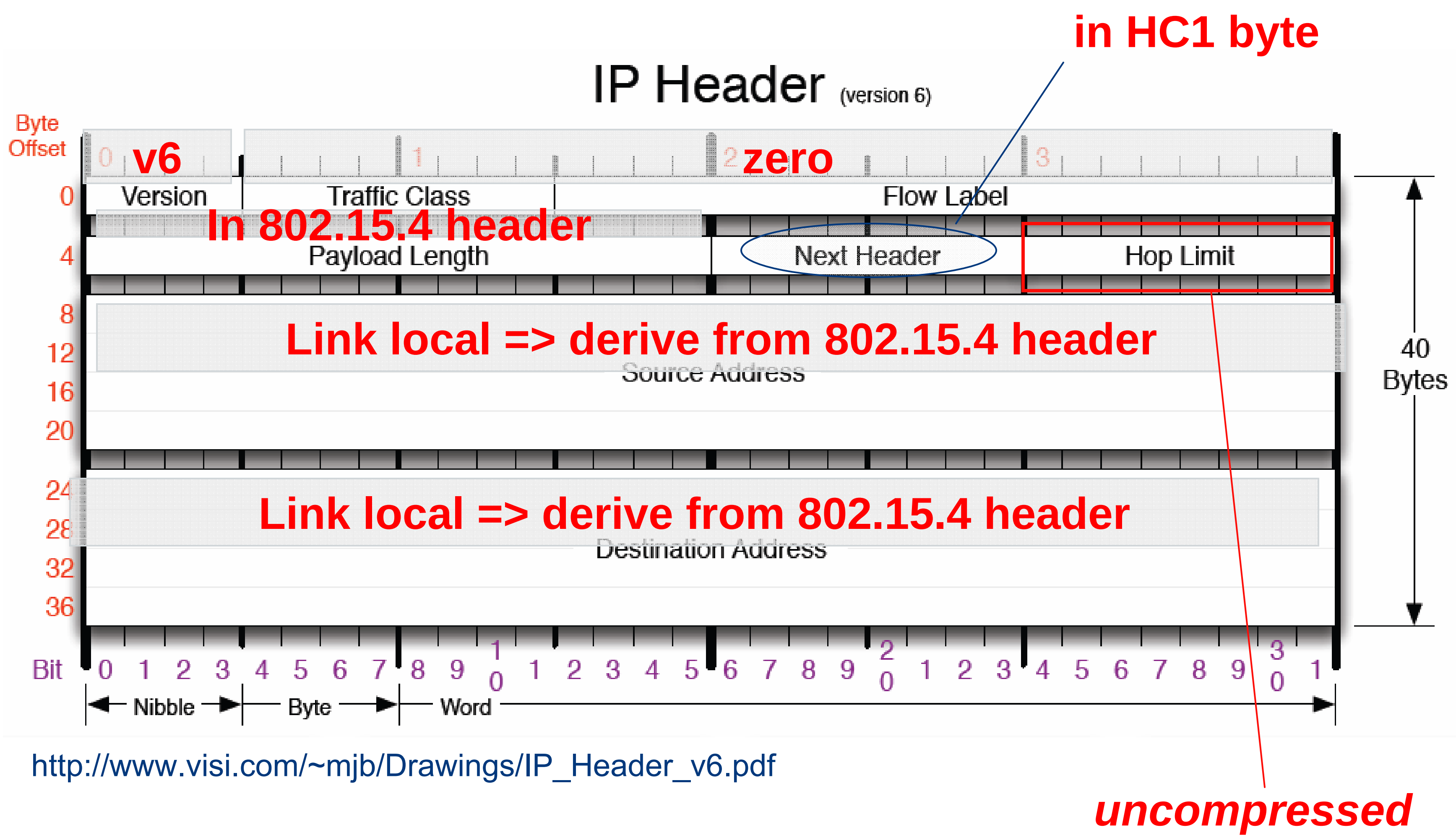
6LoWPAN Challenges



- Large IP Address & Header => 16 bit short address / 64 bit EUID
- Minimum Transfer Unit => Fragmentation
- Short range & Embedded => Multiple Hops

ARCHROCK

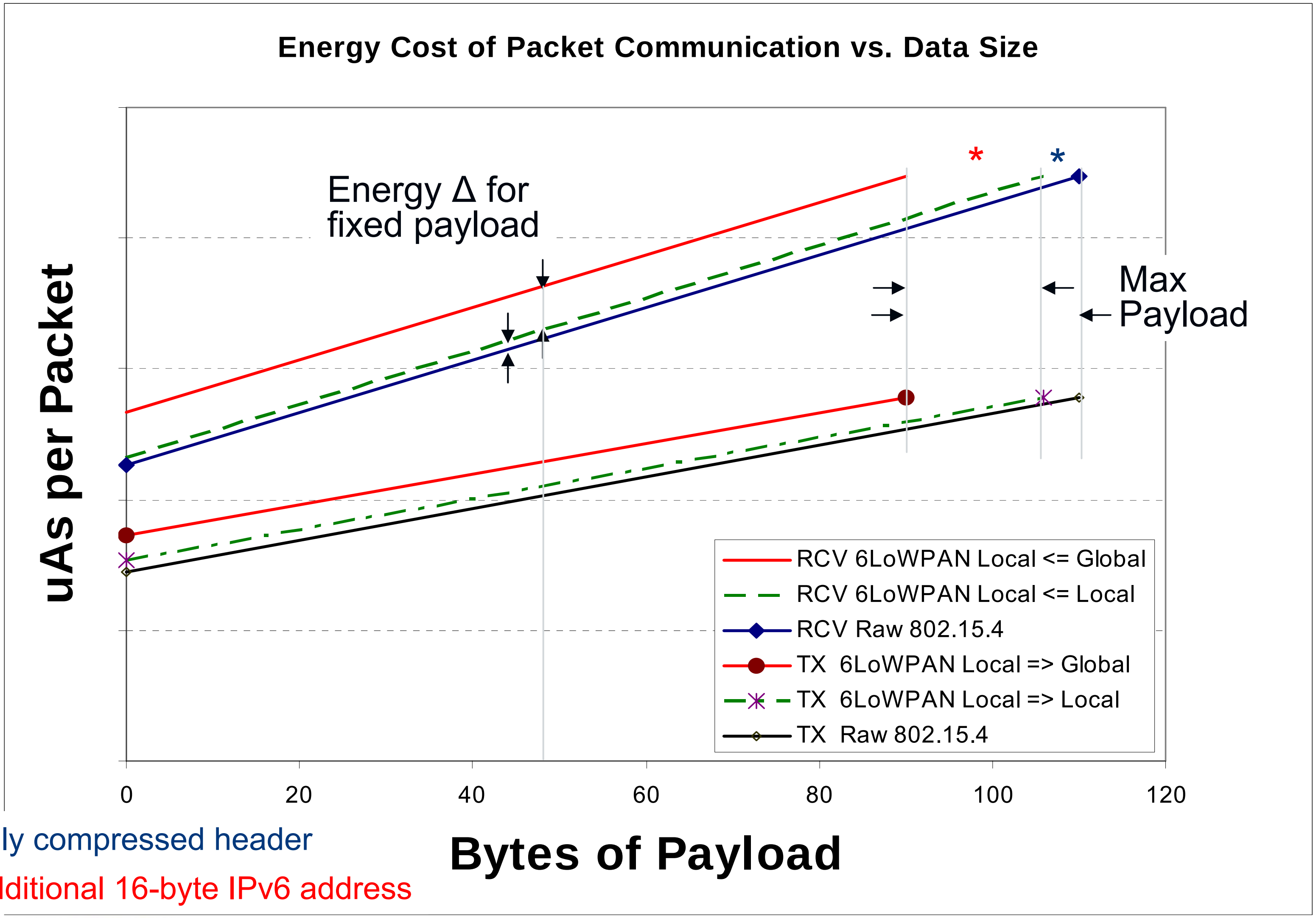
IPv6 Header Compression



- http://www.visi.com/~mjb/Drawings/IP_Header_v6.pdf

ArchRock

Low Impact of 6LoWPAN on Lifetime – Comparison to *Raw* 802.15.4 Frame

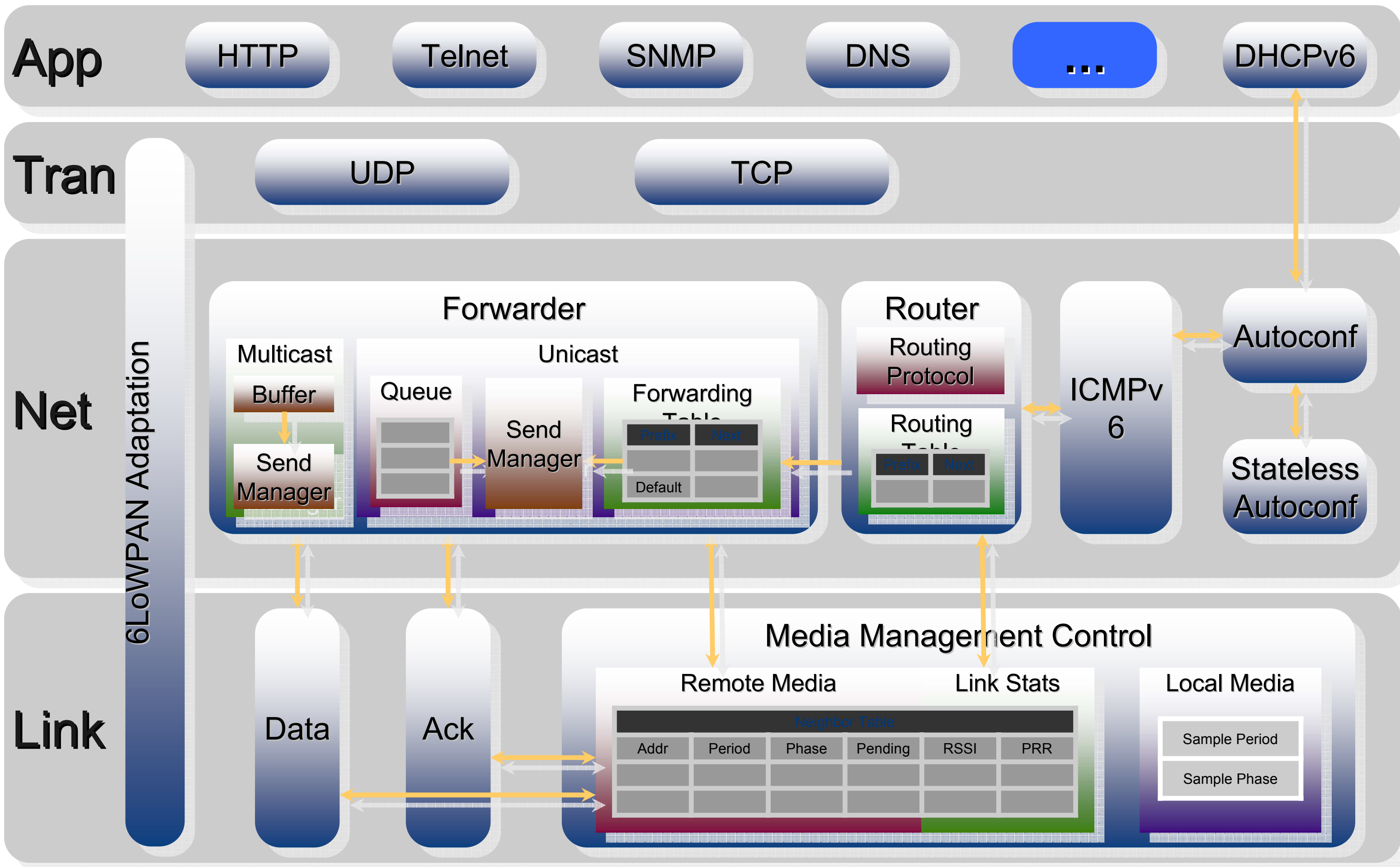


* fully compressed header

* additional 16-byte IPv6 address

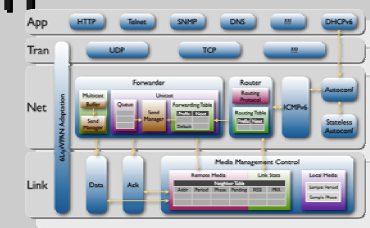
ArchRock

Complete Embedded IPv6 Stack



Embedded IPv6 in Concept

Structured Decomposition



- ✔ Retain strict modularity
  Some key cross-layer visibility

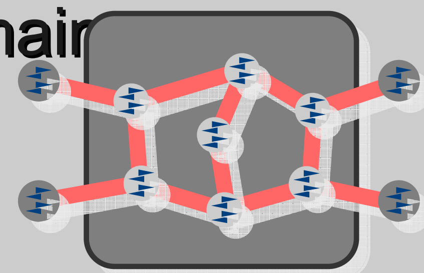
IP Link ⇒ Always On

- ✔ Retain illusion even when always off

IP Link ⇒ "Reliable"

- ✔ Retain best-effort reliability over unreliable links

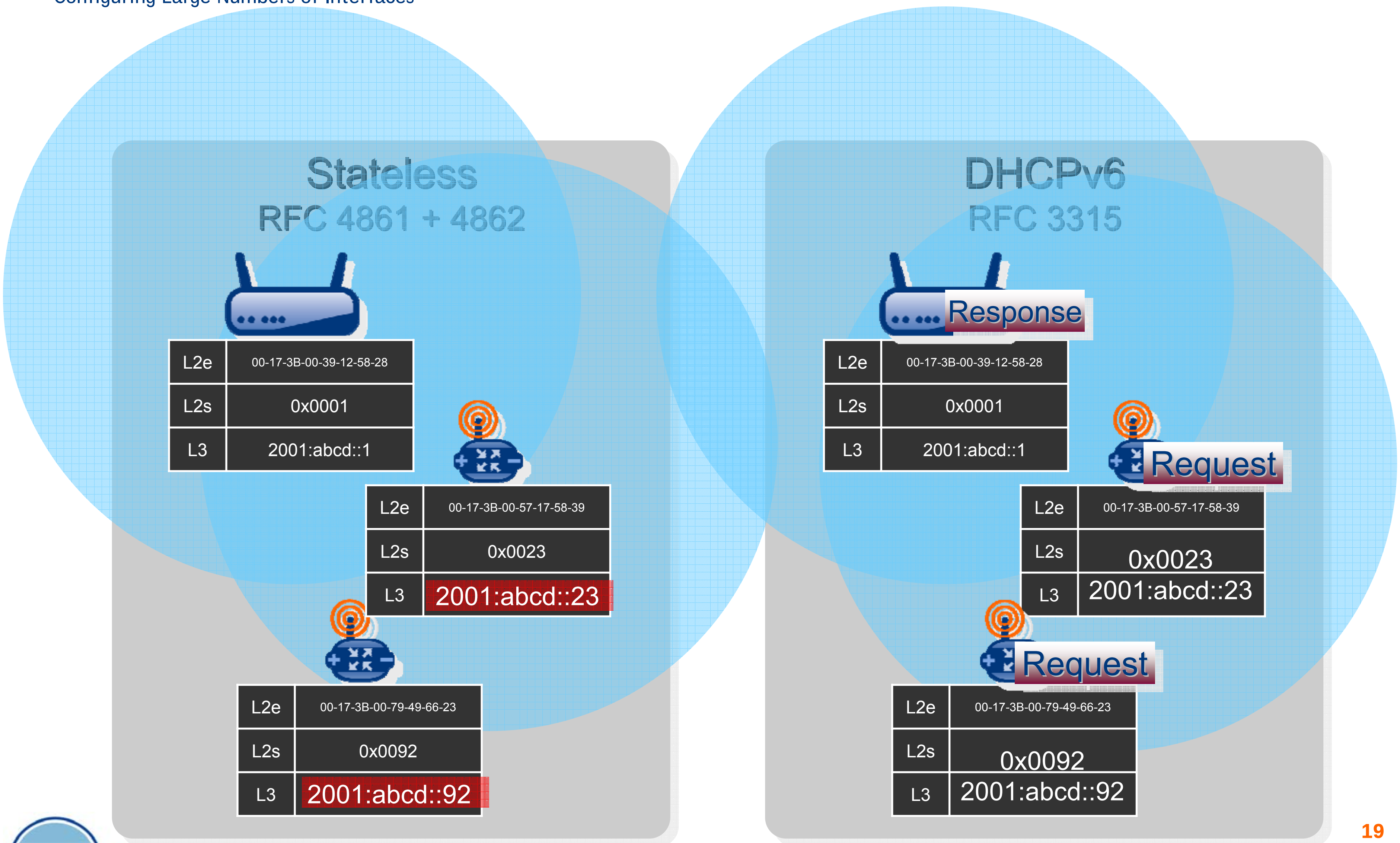
IP Link ⇒ Broadcast Domain



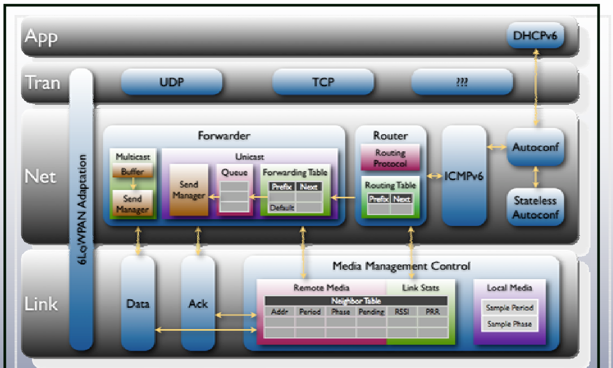
- ✘ IPv6 can support a semi-broadcast link with few changes

Autoconfiguration

Configuring Large Numbers of Interfaces



IP in TinyOS on Motes is a reality today



24038 ROM

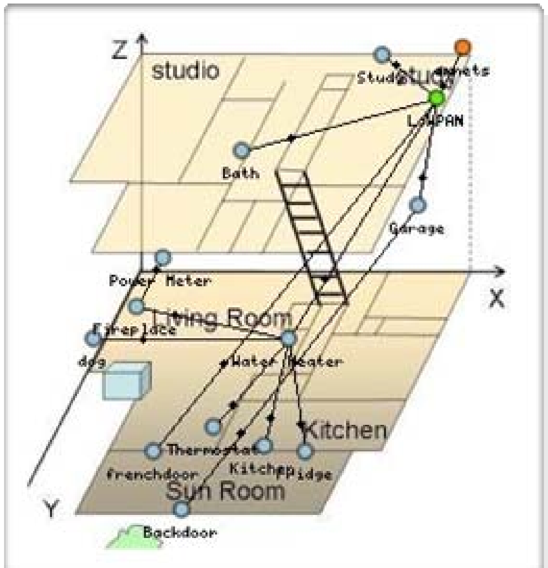
3598 RAM
(including runtime)

* Production implementation on TI msp430/cc2420

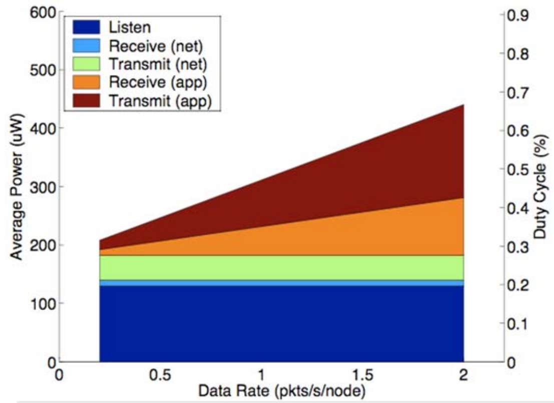
- Footprint, power, packet size, & bandwidth

	ROM	RAM
CC2420 Driver	3149	272
802.15.4 Encryption	1194	101
Media Access Control	330	9
Media Management Control	1348	20
6LoWPAN + IPv6	2550	0
Checksums	134	0
SLAAC	216	32
DHCPv6 Client	212	3
DHCPv6 Proxy	104	2
ICMPv6	522	0
Unicast Forwarder	1158	451
Multicast Forwarder	352	4
Message Buffers	0	2048
Router	2050	106
UDP	450	6
TCP	1674	50

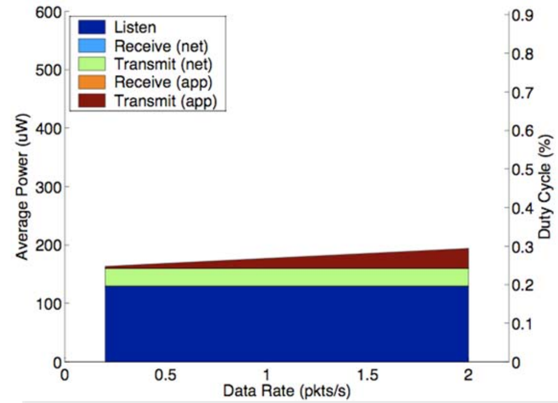
Low Power, Reliability, Scaling



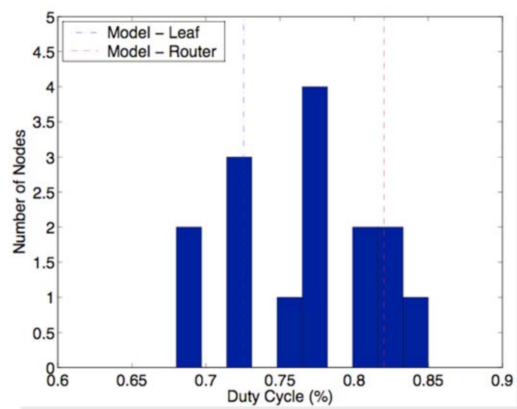
$$f_{rx} = N f_{ra} + D(f_{rr} + f_{app})$$
$$f_{txb} = f_{ra}$$
$$f_{txu} = (1 + D)(f_{rr} + f_{app})$$



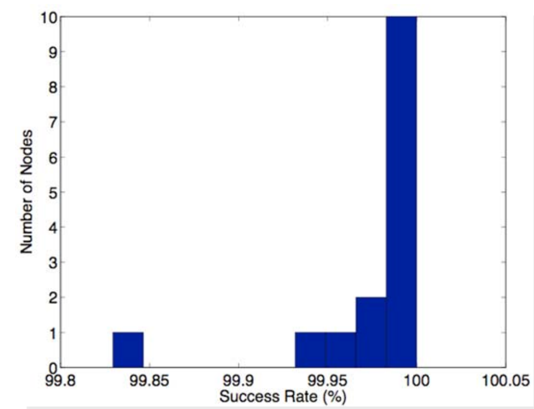
Data Rate Sensitivity
(Router)



Data Rate Sensitivity
(Edge)



Deployment Duty Cycle



Deployment Reliability

TinyOS Programming Interface

BSD Sockets ⇒ Event based UDP

```
event void Boot.booted() {call Udp.bind( 7 );  }

event void Udp.recvfrom( void *buf, uint16_t len,
                 sockaddr_in6_t *from,
                 link_metadata_t *linkmsg ) {
call Udp.sendto( buf, len, from );  }
```

ArchRock

TinyOS TCP Echo Server

BSD Sockets ⇒ Event based TCP

```
uint8_t m_buf[ BUF_SIZE ];
event void Boot.booted() {
      call Tcp.bind( 7 );
      call Tcp.listen();  }
event bool Tcp.accept( sockaddr_in6_t *to, void **sendbuf,
            uint16_t *sendbuf_size ) {
      *sendbuf = m_buf;
      *sendbuf_size = sizeof(m_buf);
      return TRUE;  }
event void Tcp.connected() {}
event uint16_t Tcp.recv( void *buf, uint16_t len ) {
      return call Tcp.send( buf, len ) == SUCCESS ? len : 0;  }
event void Tcp.acked() {}
event void Tcp.closed() {    signal Boot.booted();  }
```

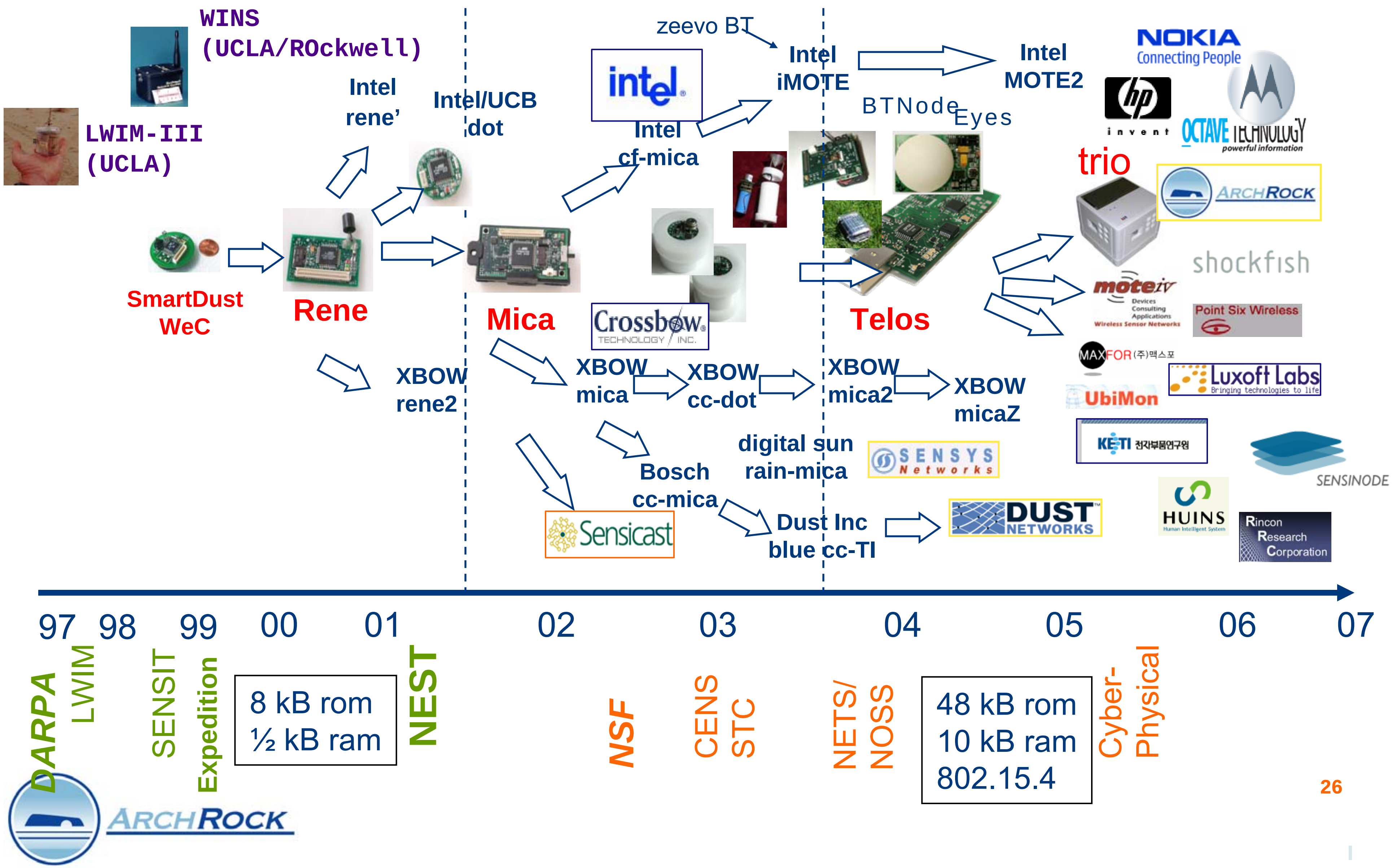
TinyOS and Industry

- TinyOS and the Berkeley Mote has always been an interplay of academia and industry
  - Academic research creates and gives to industry
  - Industry refines and gives back
- BSD license permits commercialization
  - It is not GPL
  - Preserve copyright, but take to market
- Companies give back in many forms
  - Sell products compatible with open reference
  - Hopefully, hardened and improved

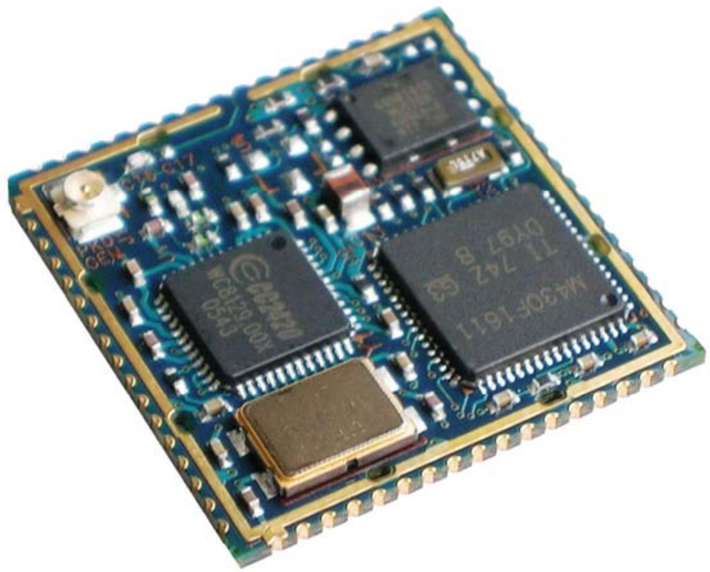
Wireless Embedded Internet - Starting Points

- http://www.eecs.berkeley.edu/~culler/WEI
  - Complete set of lectures, labs, and materials (in progress)
- http://support.archrock.com/toski
  - Evaluation version of IPv6 TinyOS Binary Kernel
  - Epic and Telosb platforms
- http://support/archrock.com/ASD
  - Arch Rock IP/6LoWPAN Software Distribution (ASD) - Atmel RZ Raven
    - Atmega + RF231
  - C kernel with a TinyOS Core

The Mote – What's Next?



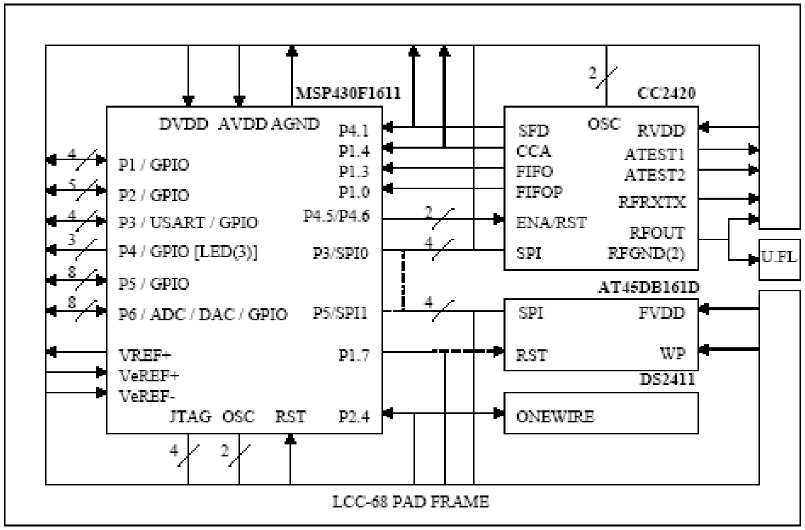
Generation 4 – EPIC



- http://www.eecs.berkeley.edu/~prabal/projects/epic/
- **Prototype => Pilot => Production**

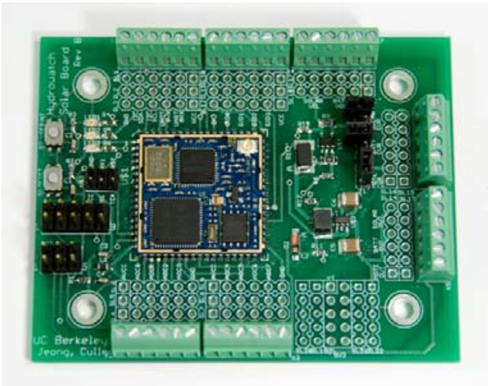
ArchRock

Epic Family



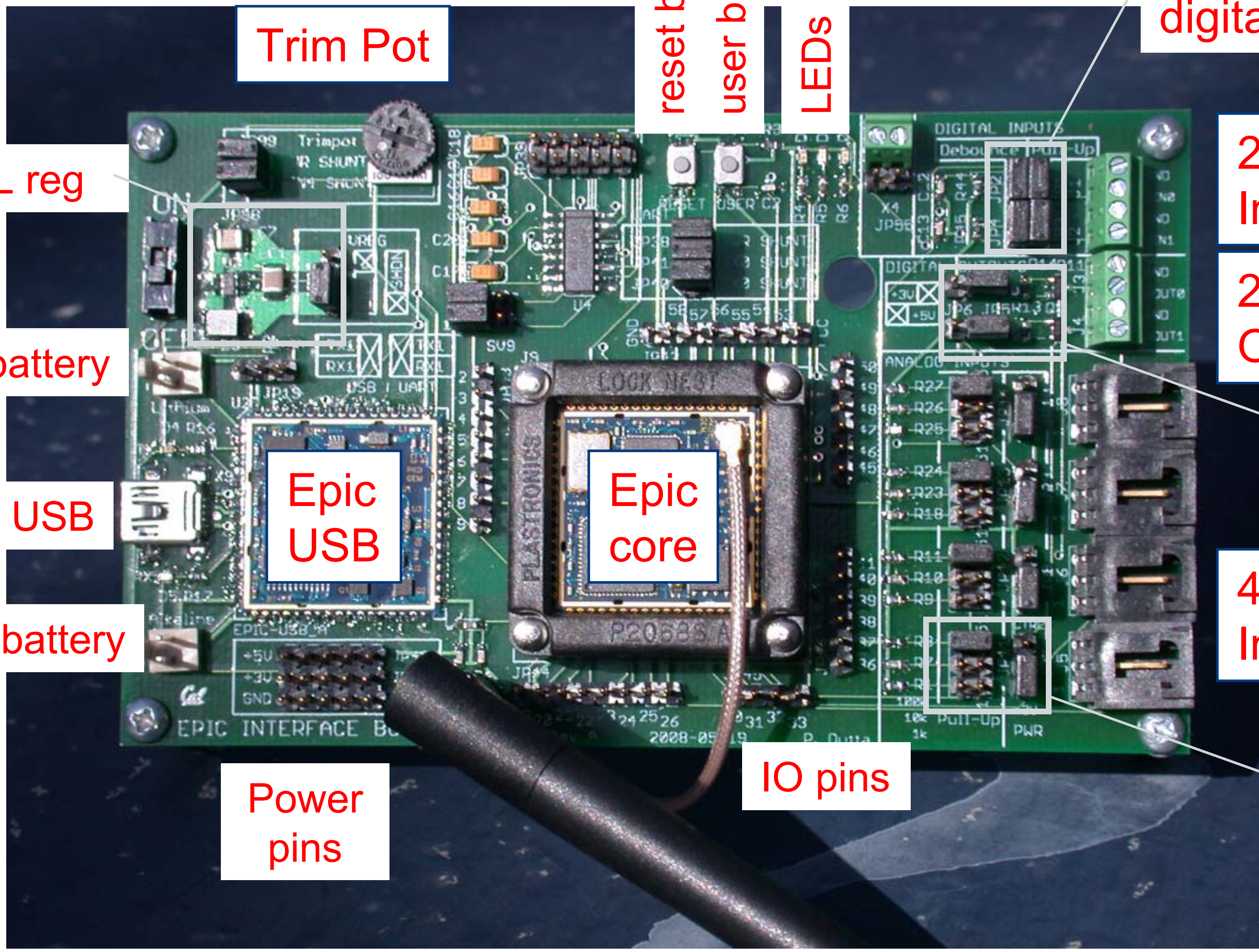
USB+Power

Storage

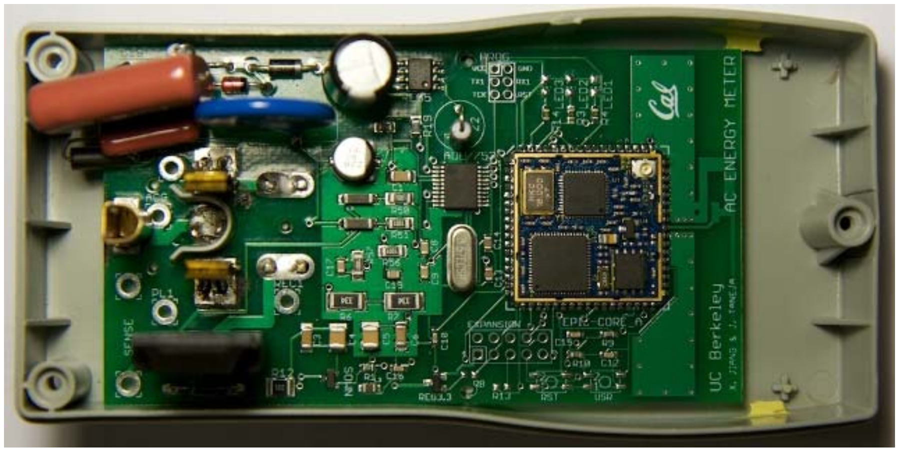


Solar + External Sensor

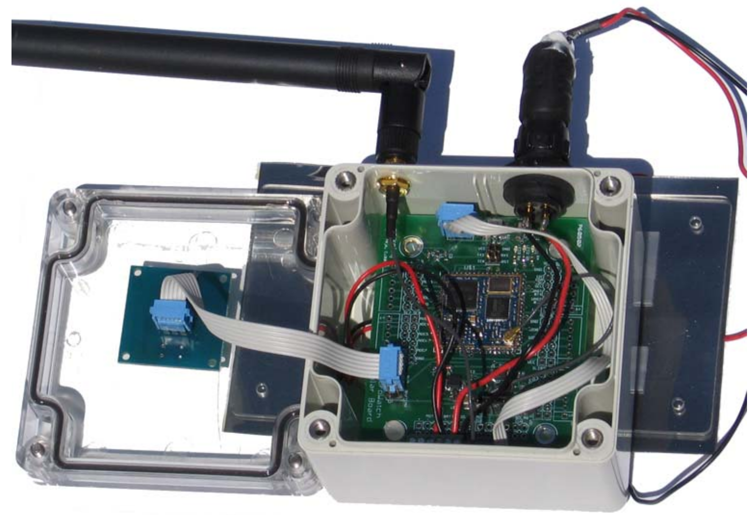
Epic Interface Board



Application Solutions



Electric Monitoring and Control



Outdoor Microclimate Monitoring

TinyOS.net – The Open Source

- TinyOS 2.0.2 is released
- TinyOS 2.1 will be a Safe Langauge
  - Compiler checks ALL pointer and Array references for Safety
  - Technology Path: UCB => MS => Utah => TinyOS Community
- Take & Give Back

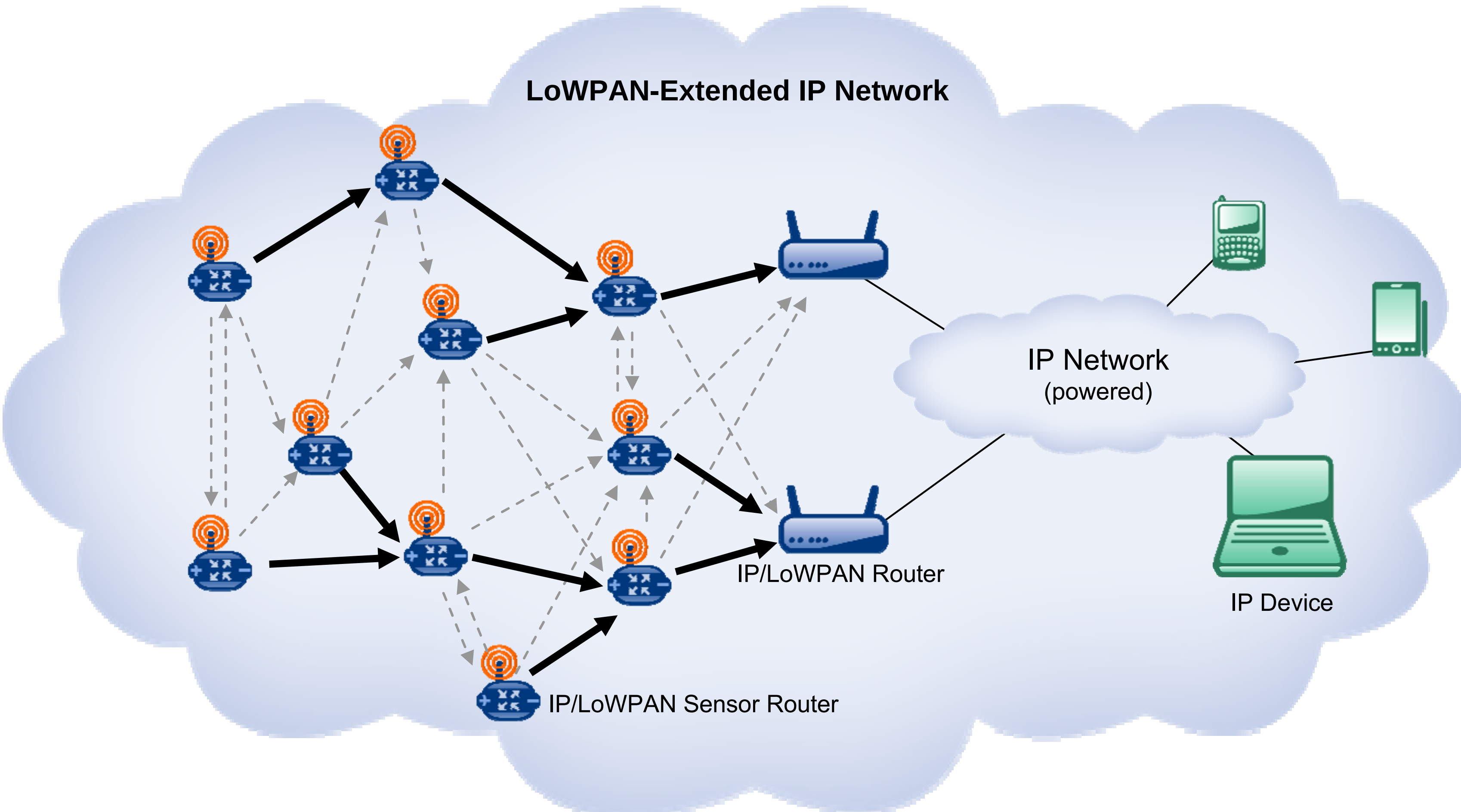
  => Contribute Code to the Community

The Next Phase

- TinyOS was invented as a framework for defining key abstractions for intelligent wireless devices embedded in the physical world.
  - Allow the right abstractions to emerge from experience
  - Hardened abstractions, platforms, community => Safety
- Advance on three fronts makes the Internet Architecture viable for this class of devices
  - Structures the problem into manageable pieces
  - Permits greater impact of high quality solutions
- New set of questions within this framework
  - LP MAC really, OS API, Cross layer visibility
  - In-network processing as overlays
- The “IP/USN” is here … today

ARCHROCK

The IP/USN


LoWPAN-Extended IP Network
IP Network
(powered)
IP/LoWPAN Router
IP Device
IP/LoWPAN Sensor Router