Communication performance of low-resource sensor motes for data-intensive applications

C. Pham

IFIP/IEEE Wireless Days 2013

November 15th, 2013

Valencia, Spain

Prof. Congduc Pham

http://www.univ-pau.fr/~cpham

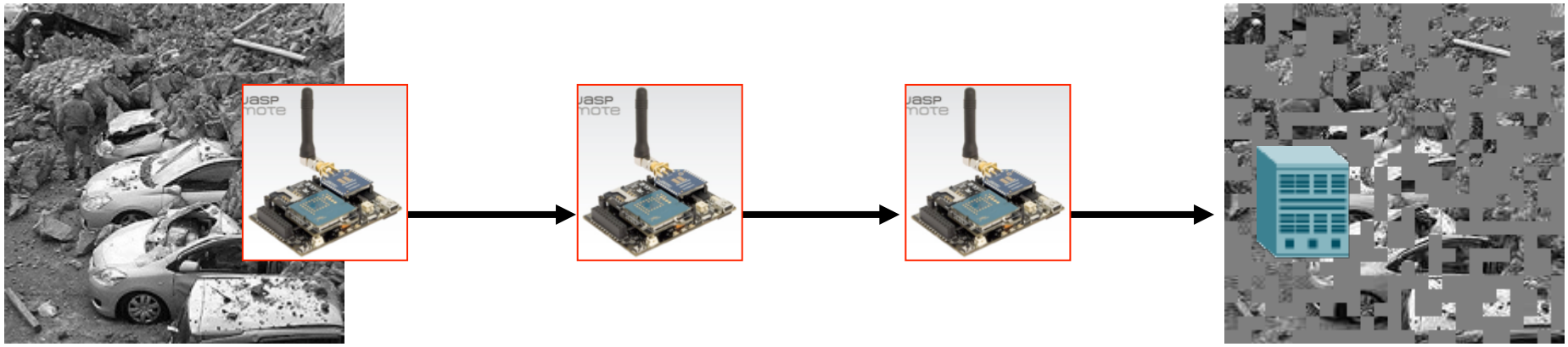
Université de Pau, France

Search & Rescue, Situation awareness

Imote2

Multimedia board

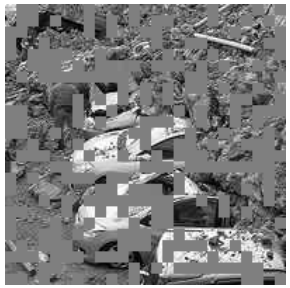
Image quality, delivery latencies,...

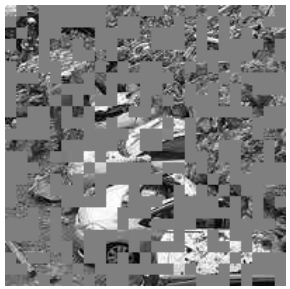


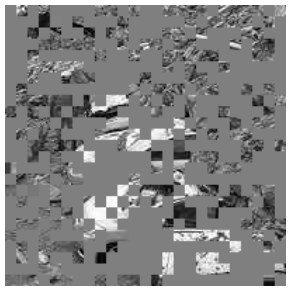

10%

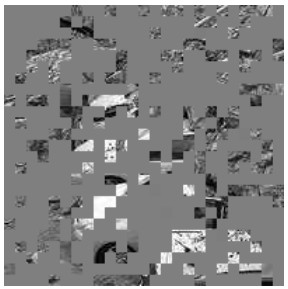

20%

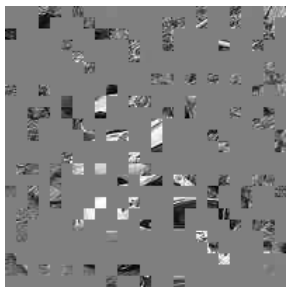

30%


40%

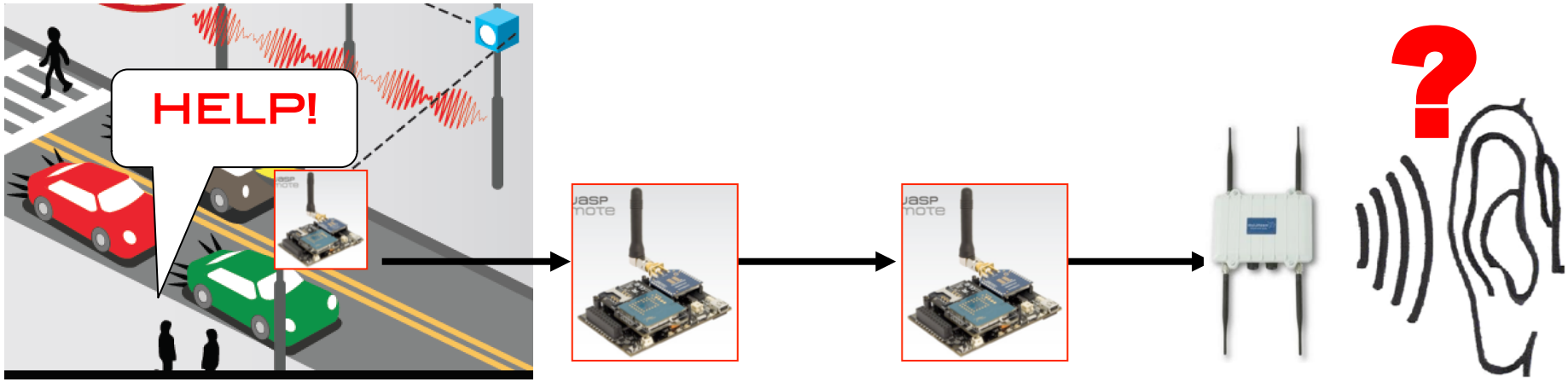

50%

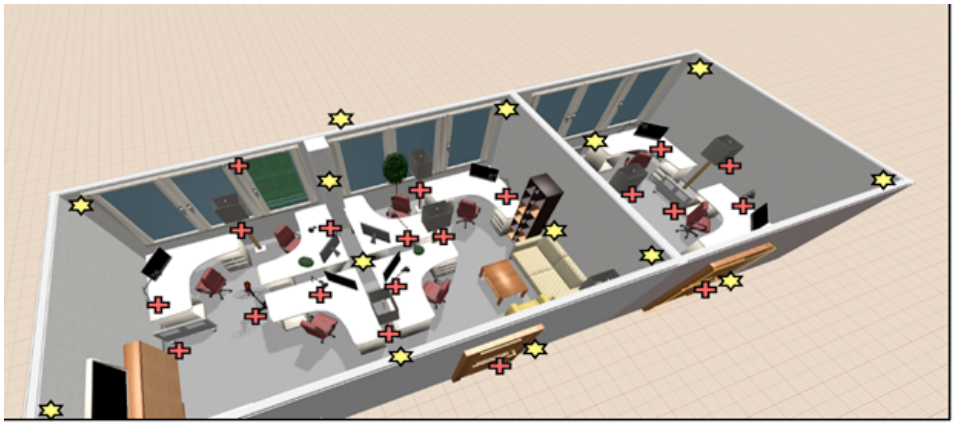

60%


70%


80%

EAR-IT: audio surveillance in SmartCities and SmartBuildings

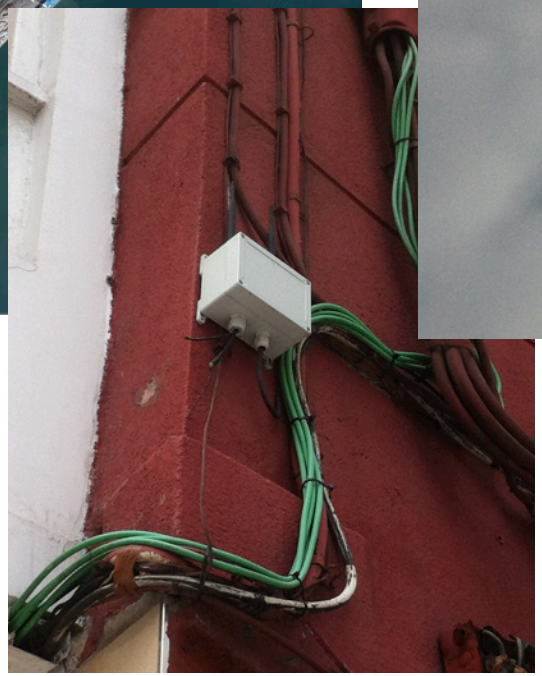




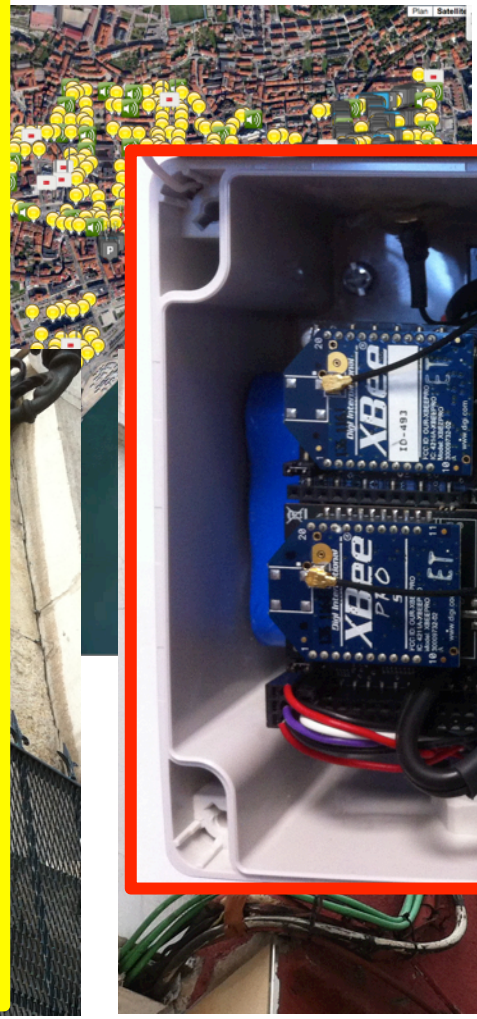
See http://www.ear-it.eu

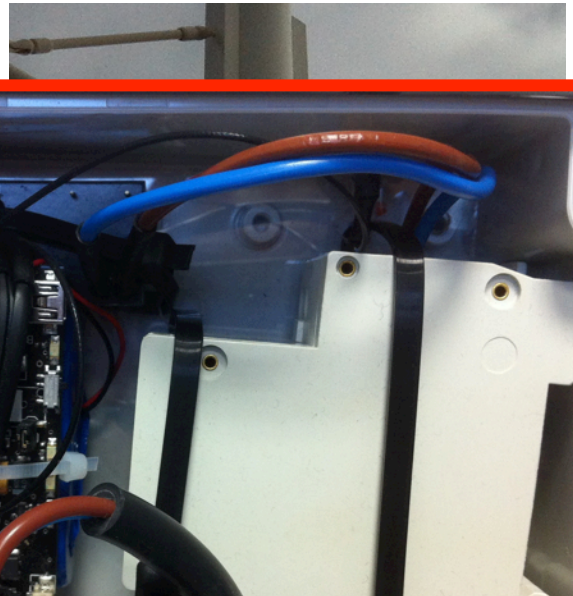
EAR-IT on SmartSantander

EAR-IT on SmartSantander

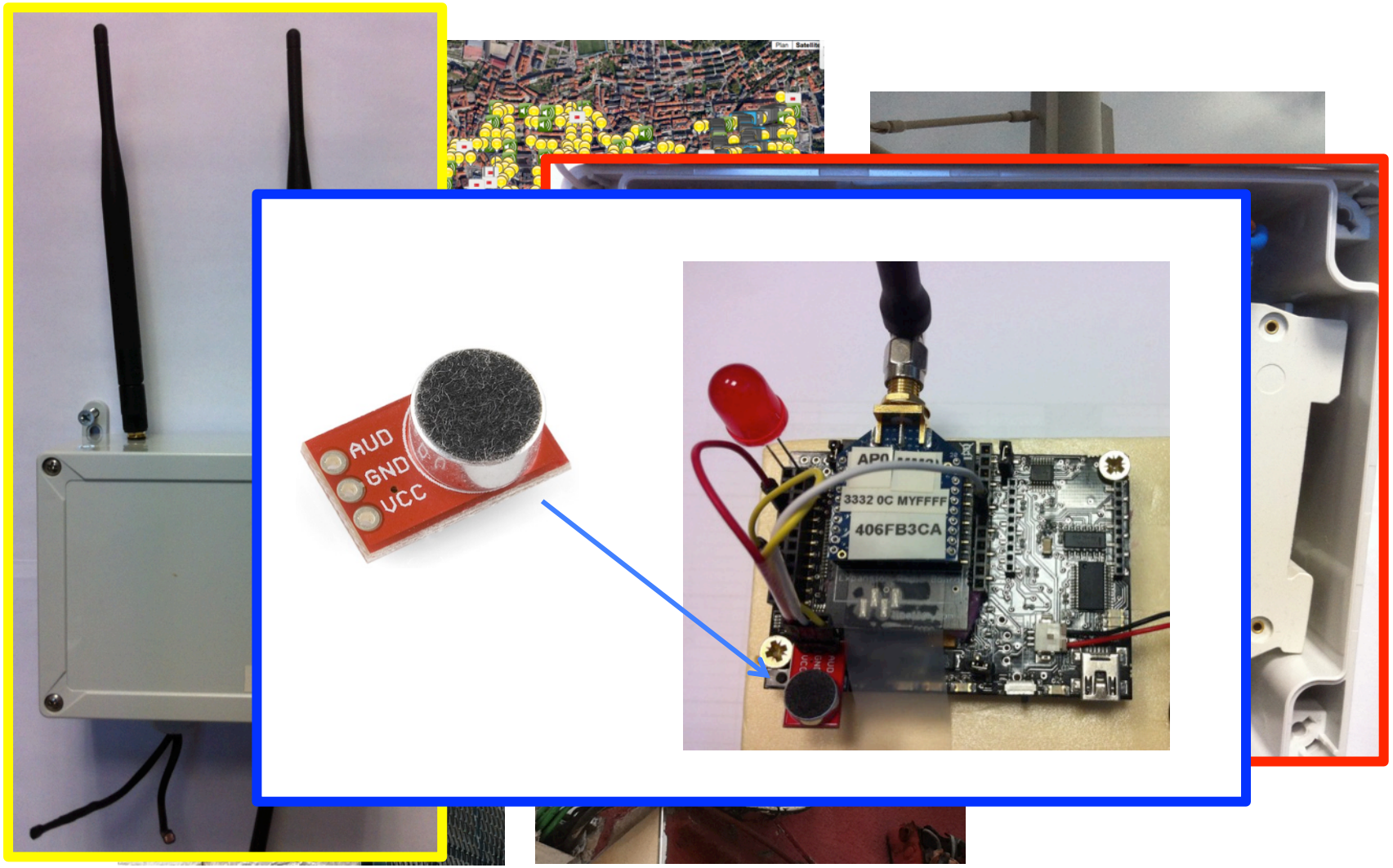


EAR-IT on SmartSantander

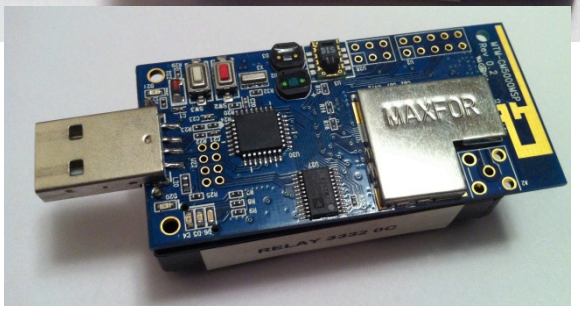


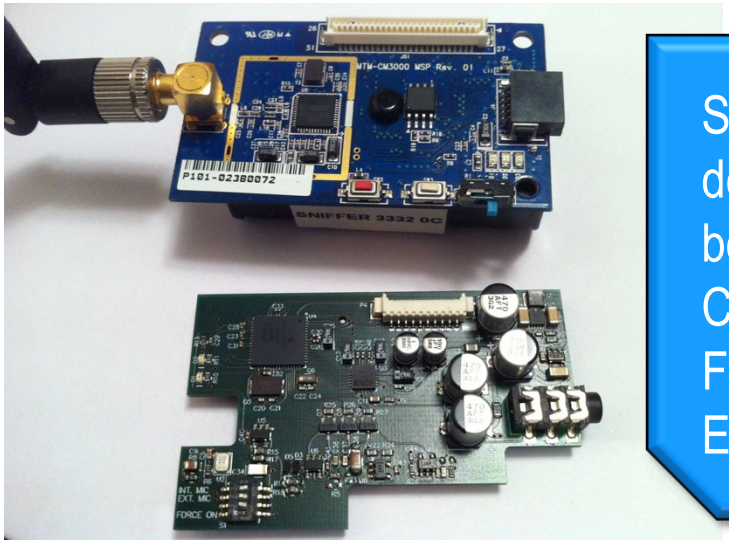


EAR-IT on SmartSantander



EAR-IT on HobNet test-bed at UNIGE



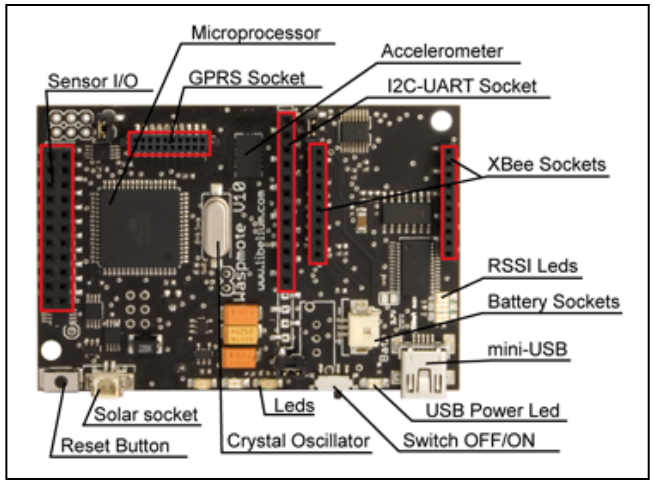


Specially designed audio board by INRIA CAIRNS & Feichter Electronics

dsPIC33 with 8kbps speex real-time encoder

EAR-IT IoT node

libelium
comunicaciones inalámbricas distribuidas



WaspMote

8MHz Atmega1281
8kB SRAM, 128kB Flash
Xbee radio

advanticsys

AdvanticSys CM5000
TelosB-like mote

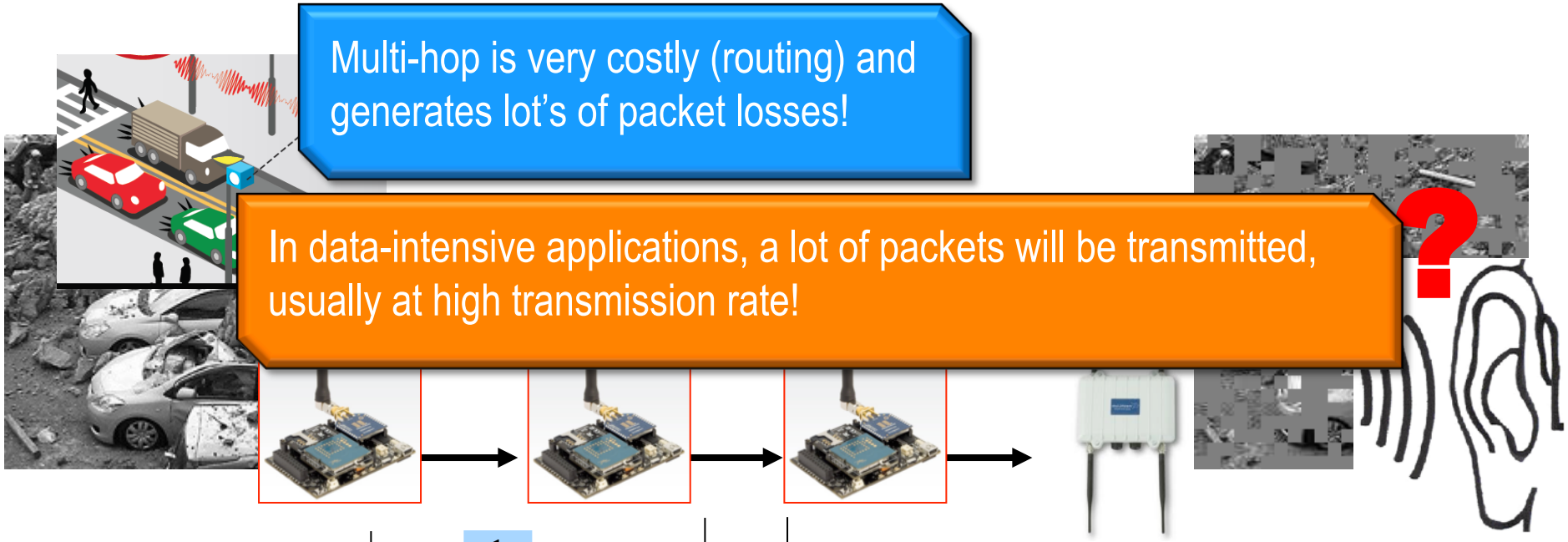
8Mhz MSP430F1611
48K flash, 10K RAM
CC2420 radio

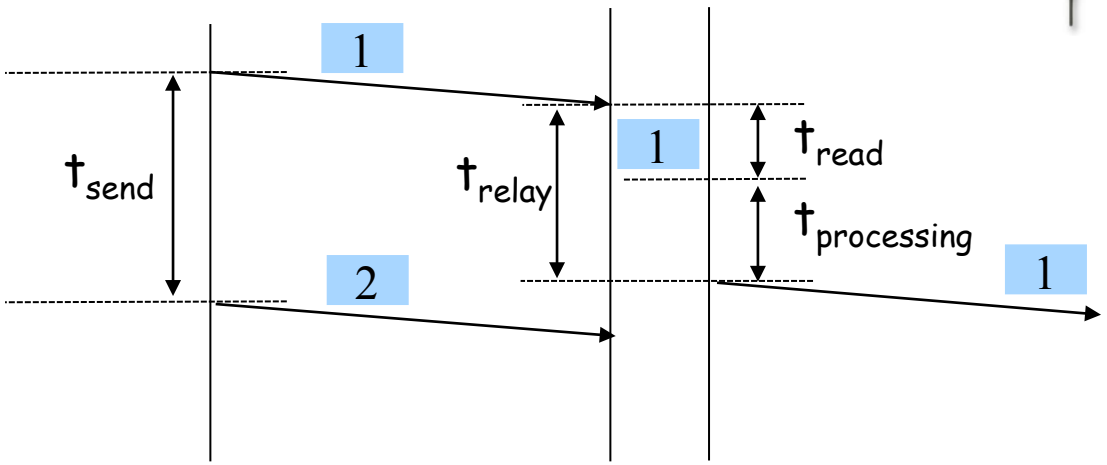
IMAGES ARE FROM LIBELIUM COMPANY

Multi-Hop Packet Forwarding

Multi-hop is very costly (routing) and generates lot's of packet losses!

In data-intensive applications, a lot of packets will be transmitted, usually at high transmission rate!

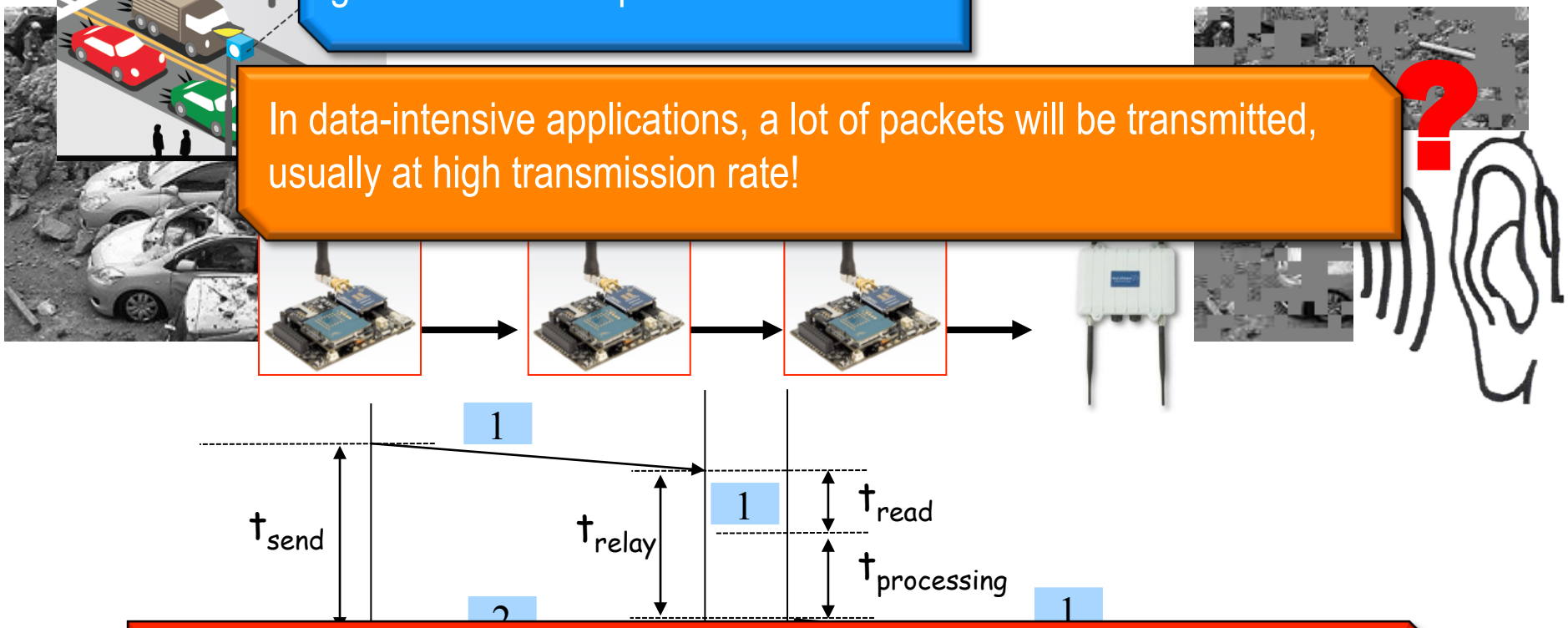




Multi-Hop Packet Forwarding

Multi-hop is very costly (routing) and generates lot's of packet losses!

In data-intensive applications, a lot of packets will be transmitted, usually at high transmission rate!

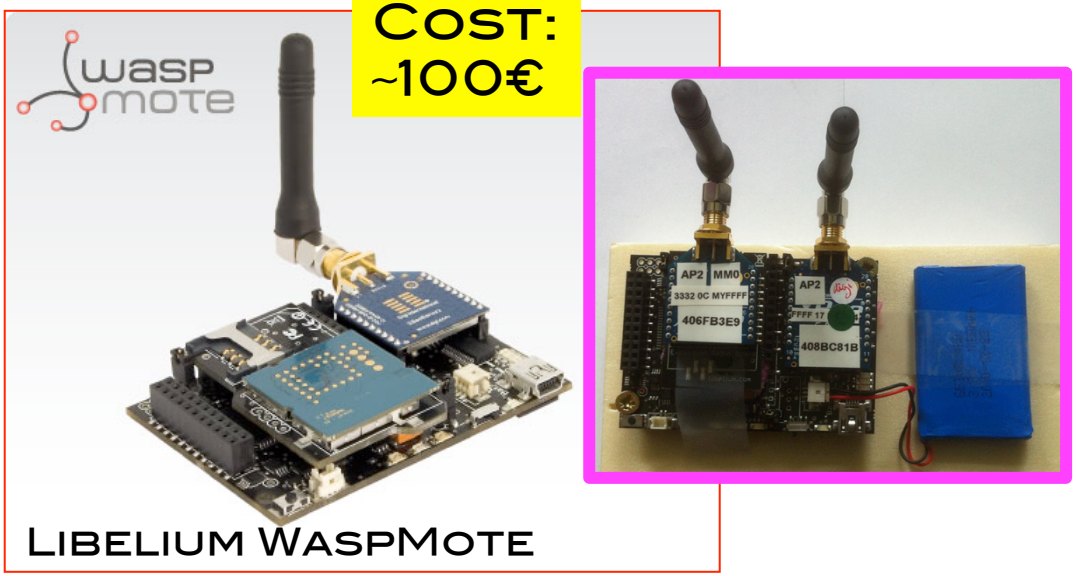


What level of performances can we expect?

Mass-market sensors

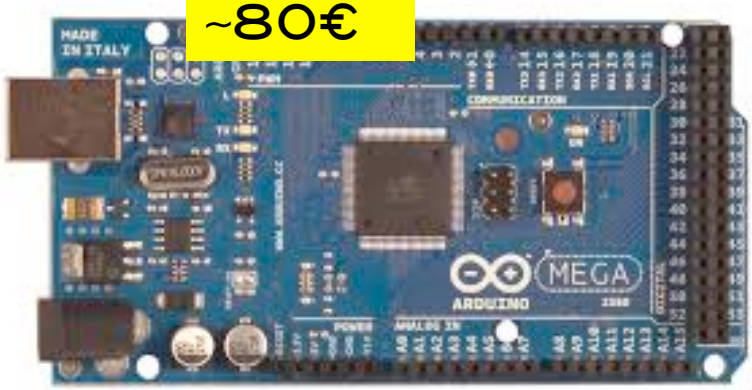
8MHz Atmega1281
8kB SRAM, 128kB Flash
Xbee radio

Cost:
~100€



Libelium WaspMote

Cost:
~80€

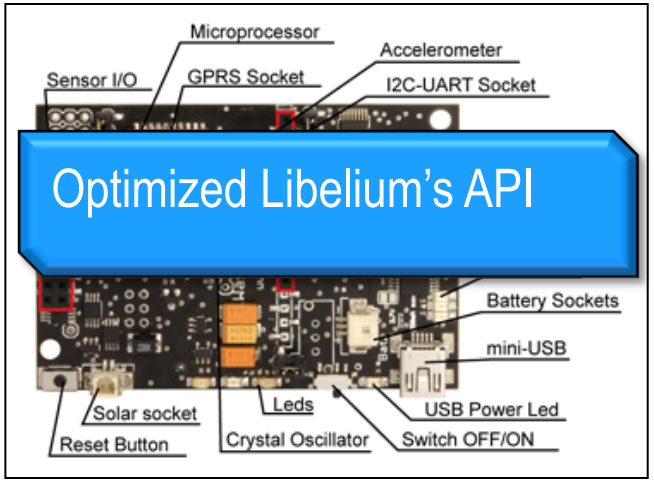


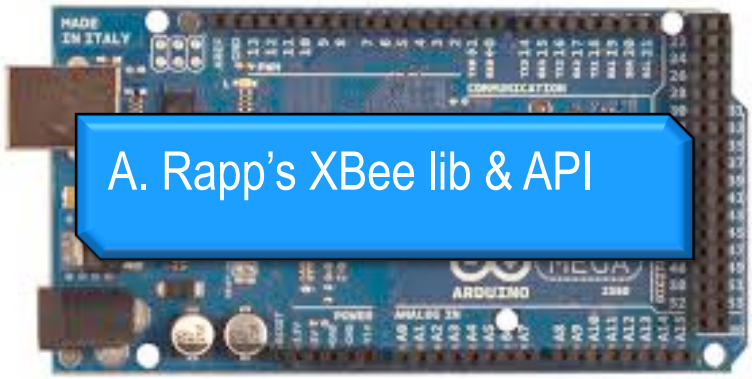
Arduino Mega2560

16MHz Atmega1281
8kB SRAM, 128kB Flash
Xbee radio

Sensor's HW&SW

Libelium WaspMote



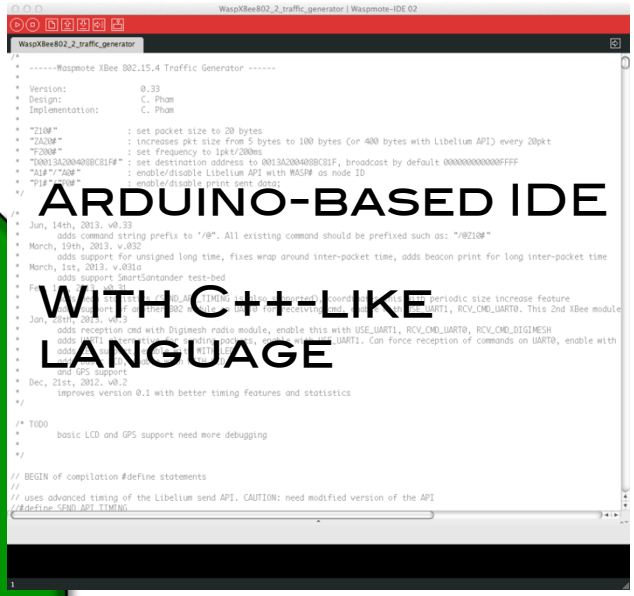


Arduino Mega2560

UART-based connection to micro-controller

Default speed is usually 38400 bauds

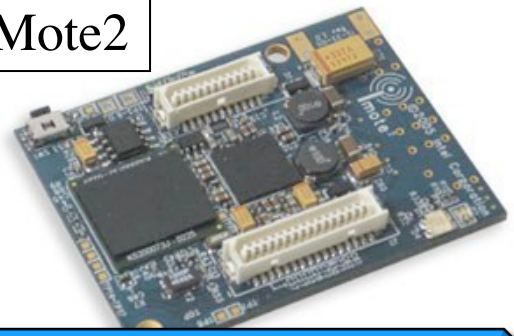
Higher baud rate are possible but…



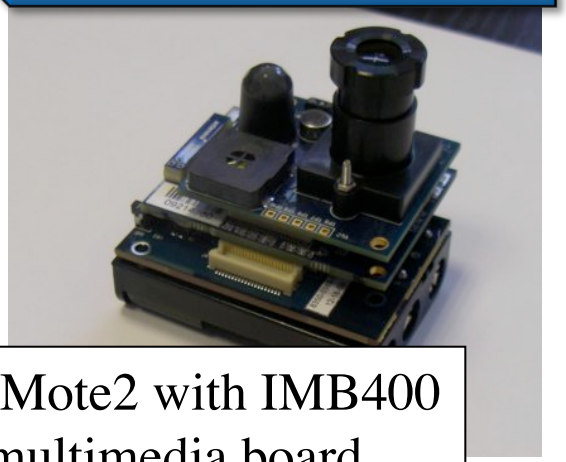
XBee 802.15.4

« Academic » sensors

iMote2



13-416MHz PXA271 Xscale
Wireless MMX DSP
256kB SRAM, 32MB Flash,
32MB SDRAM
CC2420 radio



iMote2 with IMB400
multimedia board

MICAz

8MHz Atmega128L
4kB SRAM, 128kB Flash
CC2420 radio

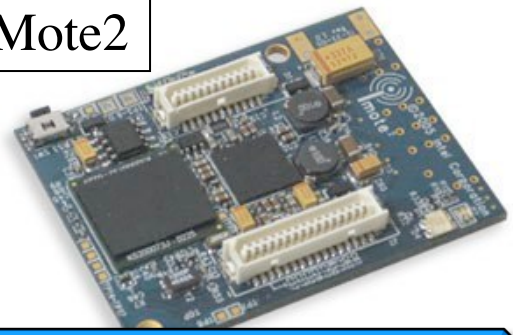
AdvanticSys CM5000 & CM3000
TelosB-like mote

TelosB

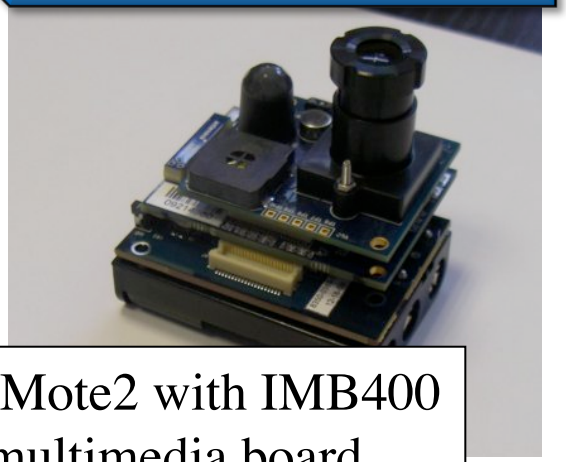
8Mhz MSP430F1611
10K SRAM, 48K flash
CC2420 radio

« Academic » sensors

iMote2



13-416MHz PXA271 Xscale
Wireless MMX DSP
256kB SRAM, 32MB Flash,
32MB SDRAM
CC2420 radio



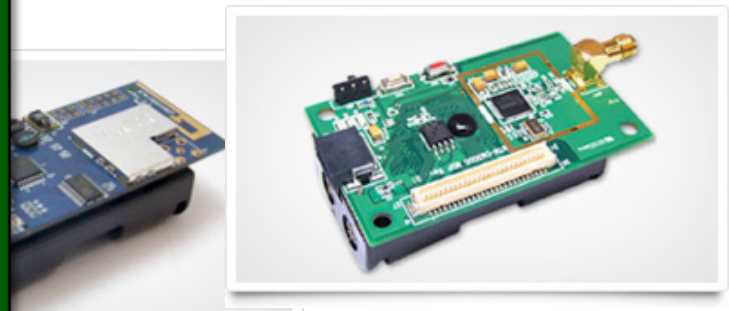
iMote2 with IMB400
multimedia board

8MHz Atmega128L
4kB SRAM, 128kB Flash
CC2420 radio

Radio module
CC2420 is
connected
through SPI bus

SPI speed is in
the order of
several
hundredth kbps

TelosB

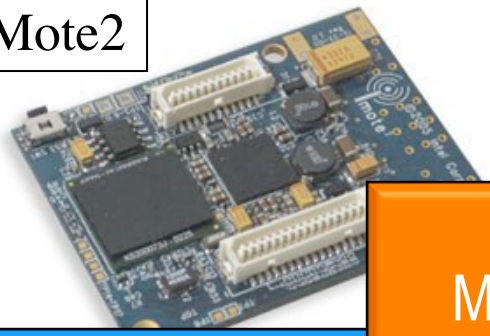


AdvanticSys CM5000 & CM3000
TelosB-like mote

8Mhz MSP430F1611
10K SRAM, 48K flash
CC2420 radio

« Academic » sensors

iMote2



8MHz Atmega128L
4kB SRAM, 128kB Flash
[illegible]420 radio

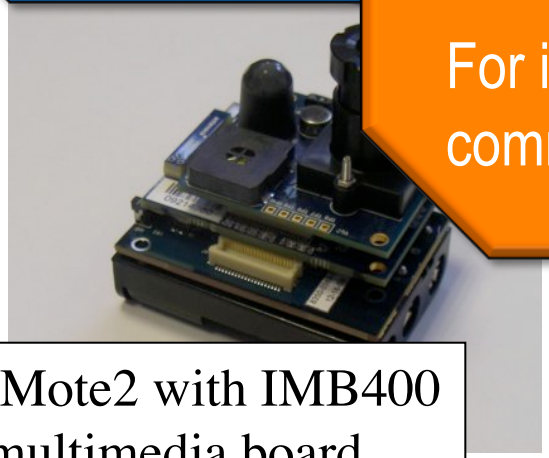
13-416MHz PXA271
Wireless MMX DSP
256kB SRAM, 32MB
32MB SDRAM
CC2420 radio

Motes are programmed under the TinyOS operating system & lib

For MicaZ and TelosB we use TKN154 communication stack

For iMote2 we use IEEE154 communication stack

s CM5000 & CM3000
mote



iMote2 with IMB400
multimedia board

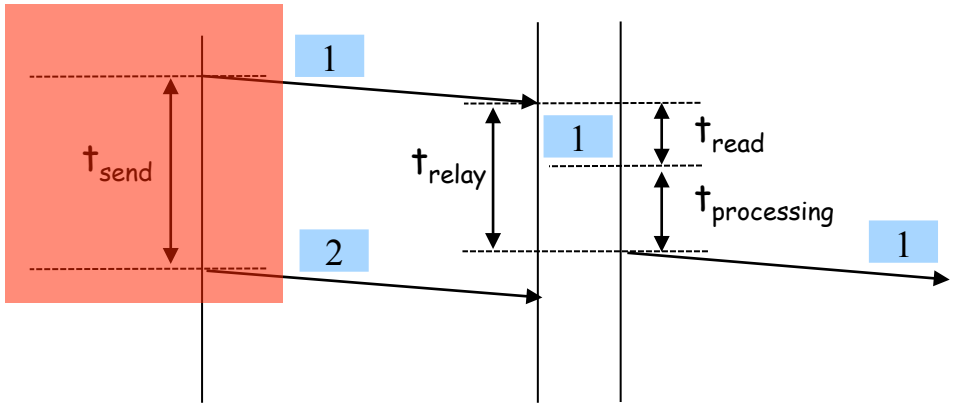
TelosB

8Mhz MSP430F1611
10K SRAM, 48K flash
CC2420 radio

Communication performances

- Application level performances depends on OS, API, hardware architecture
- Usually much lower than radio performances
- What are minimum latencies & max. throughput?
  - For sending?
  - For receiving?
  - For relaying?

Sending performances



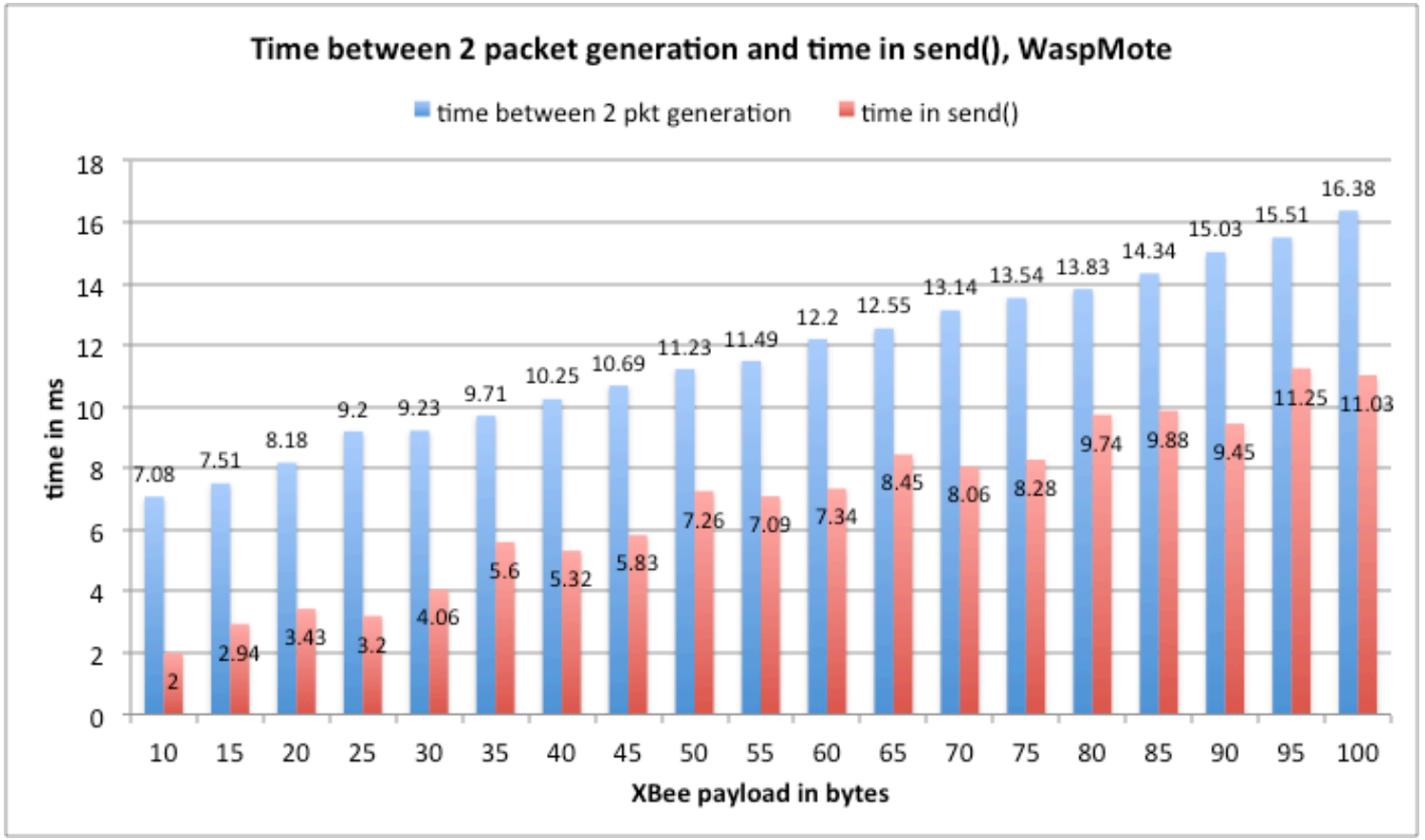
Traffic Generator

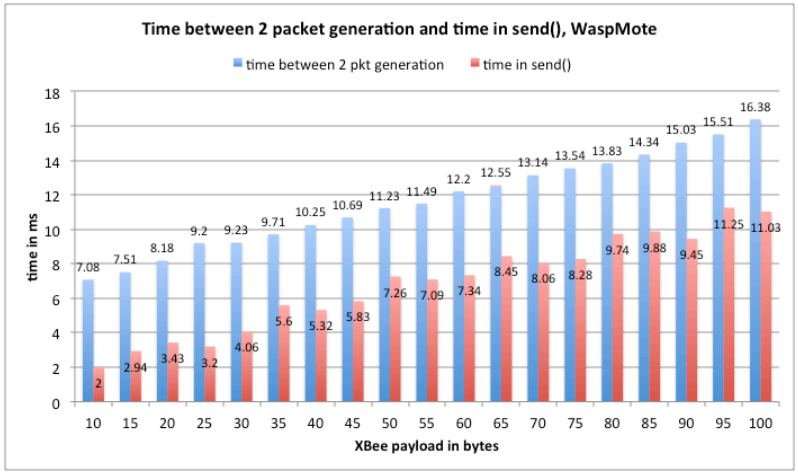
```
void loop() {
        T0;
        L0=T0;
        ...
        T1;
        send(buf);
        T2;
        ...
}
```

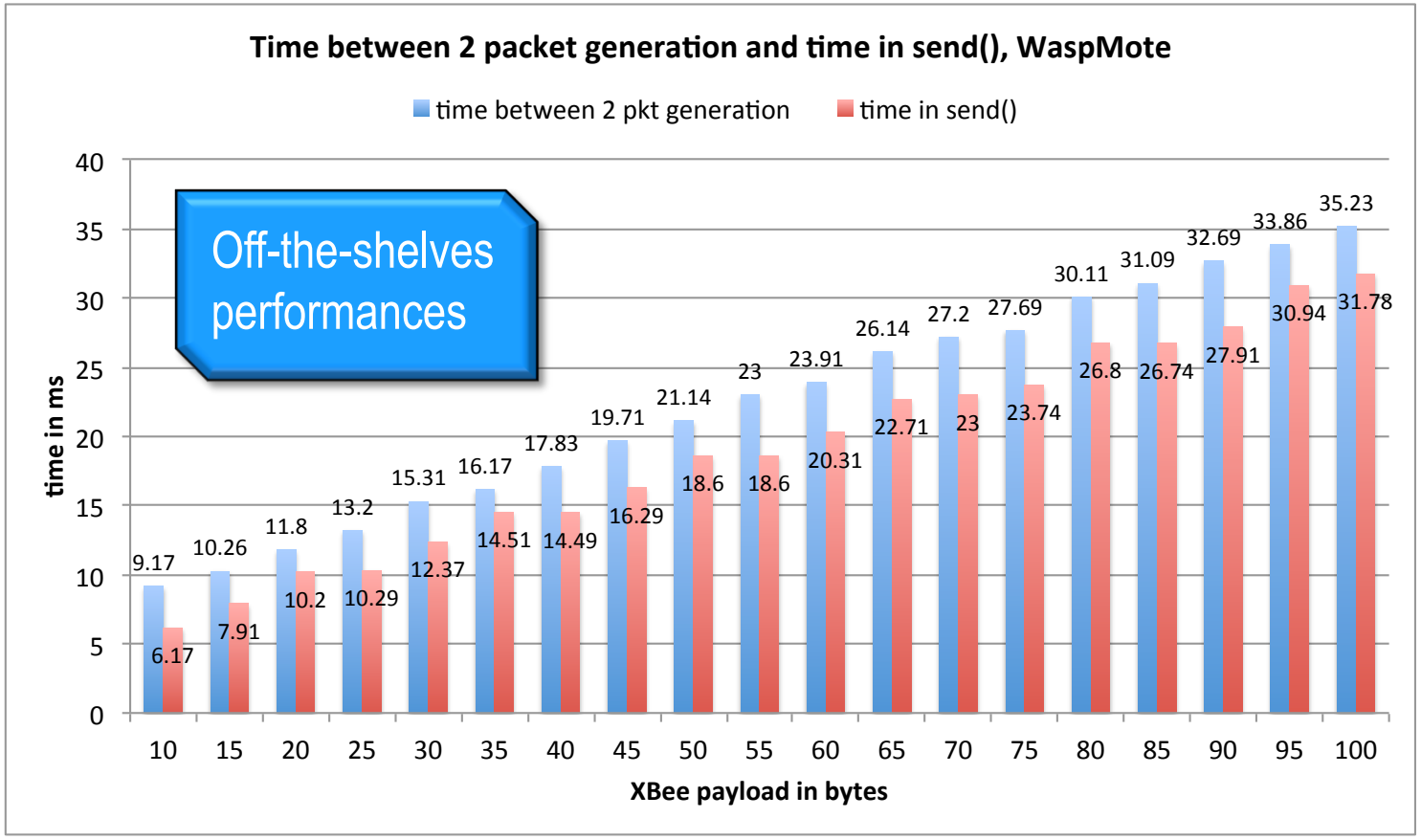
« Time in send() » is T2-T1
« Time between 2 pkt generation » is T0-L0
Time resolution is millisecond
Minimum data manipulation

Measure the time in various part of API `send()` when possible.

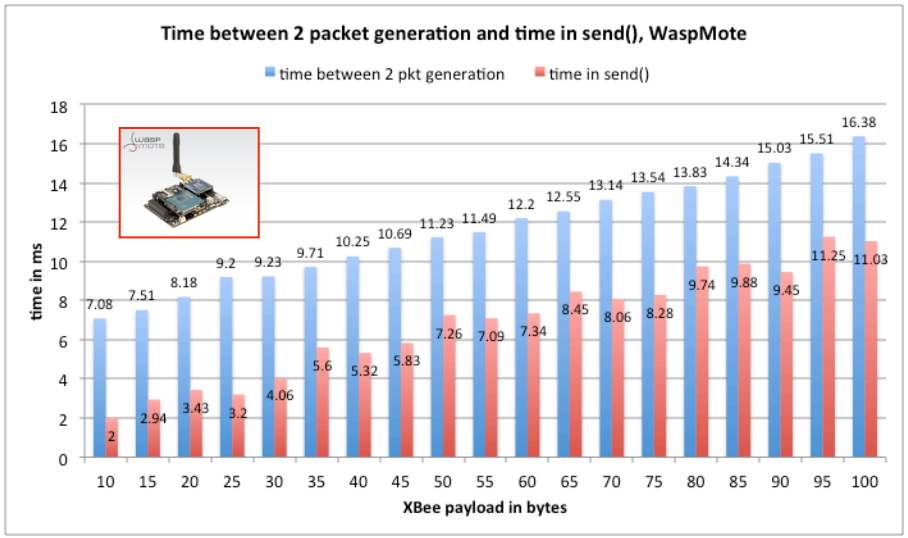
Sending performances

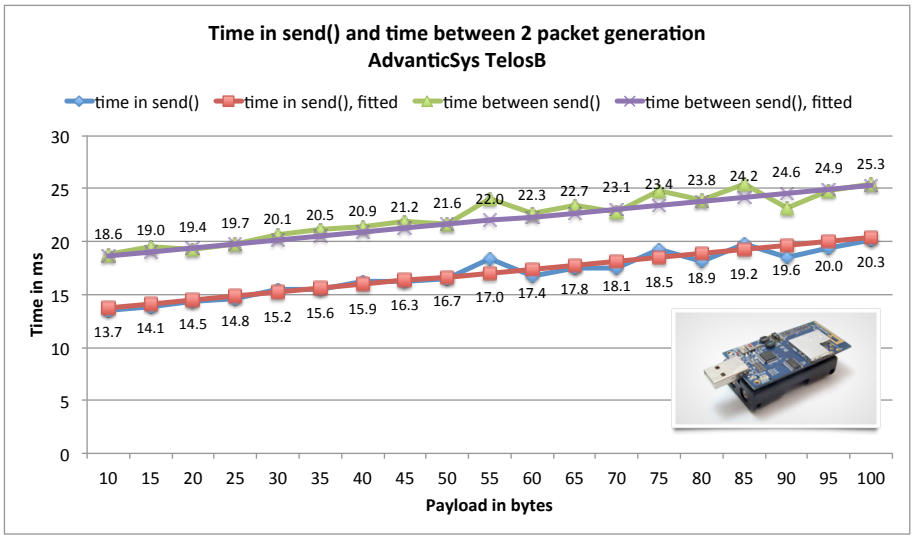


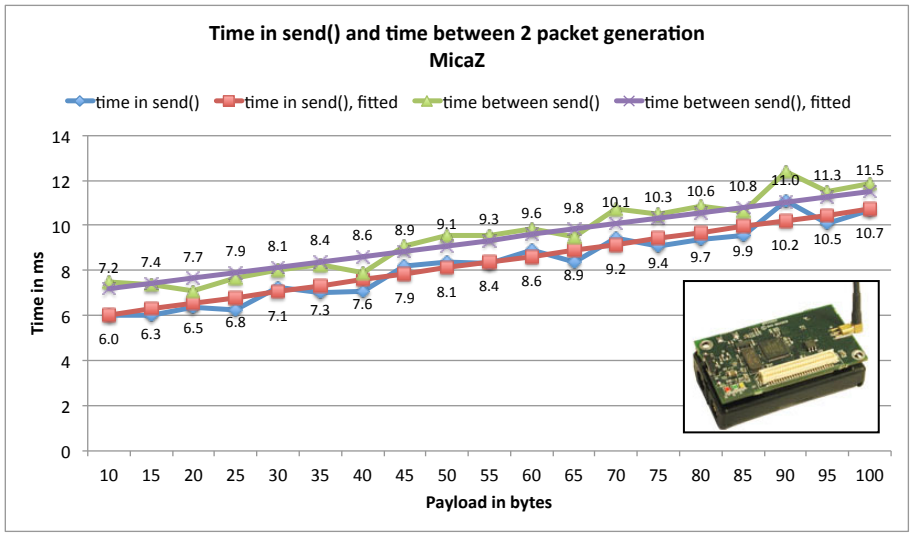

Time between 2 packet generation and time in send(), WaspMote
time between 2 pkt generation
time in send()
time in ms
XBee payload in bytes

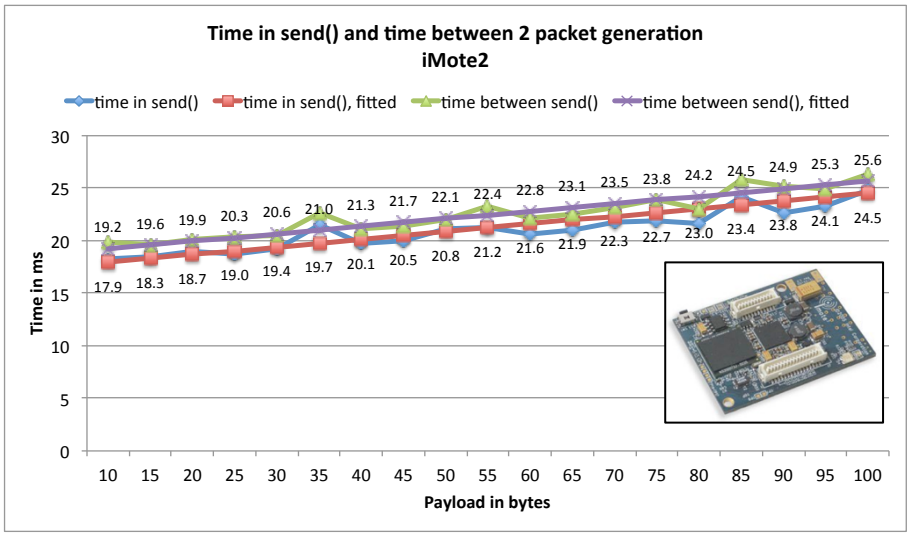

Time between 2 packet generation and time in send(), WaspMote
time between 2 pkt generation
time in send()
Off-the-shelves performances
time in ms
XBee payload in bytes

Sending performances: comparison

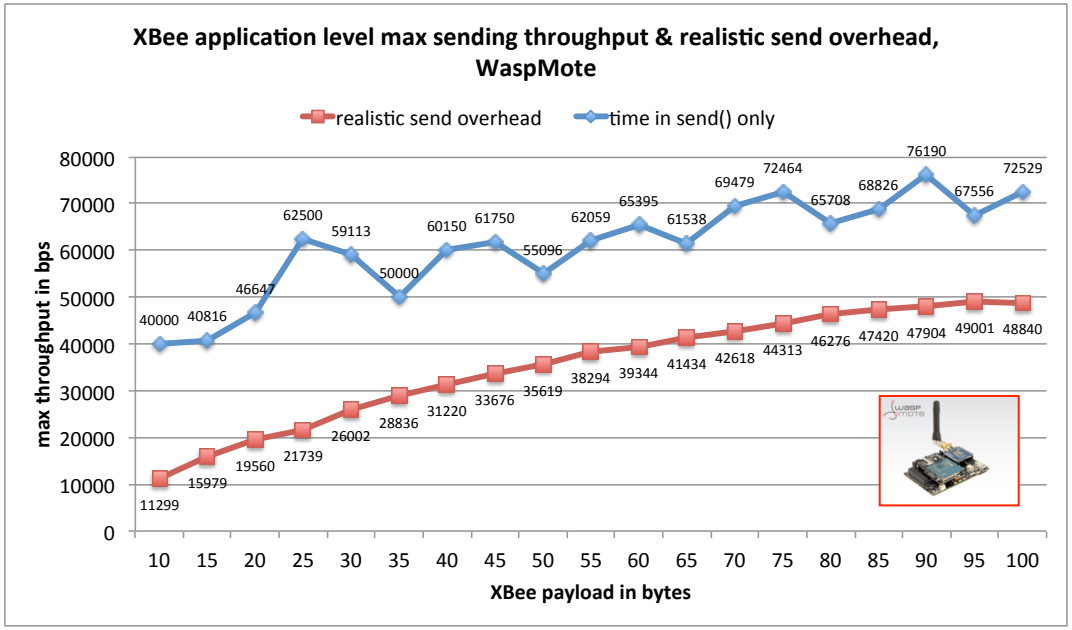




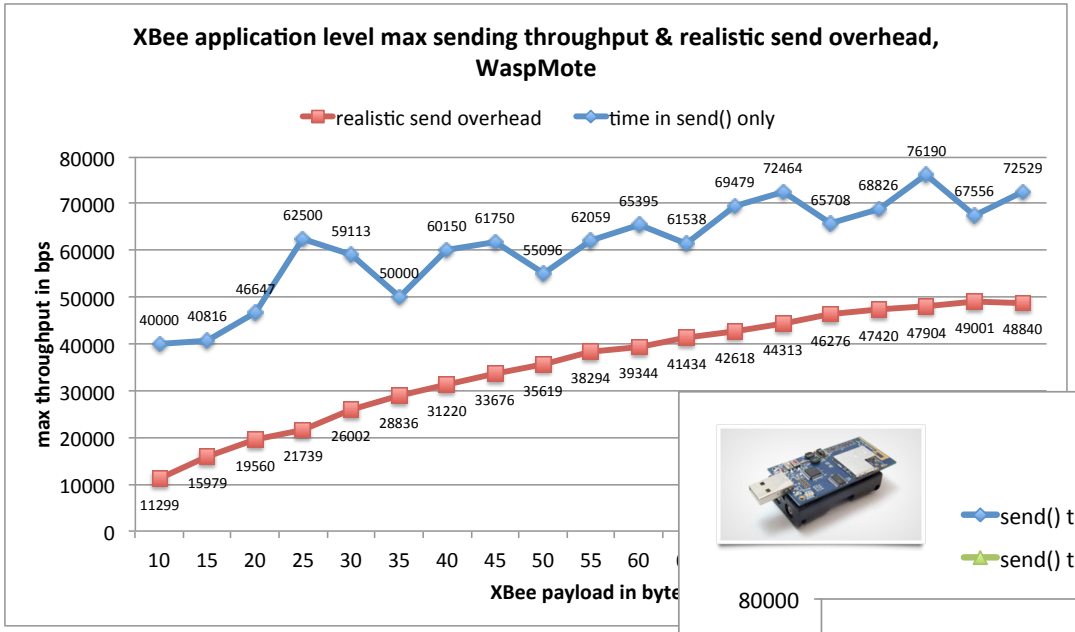


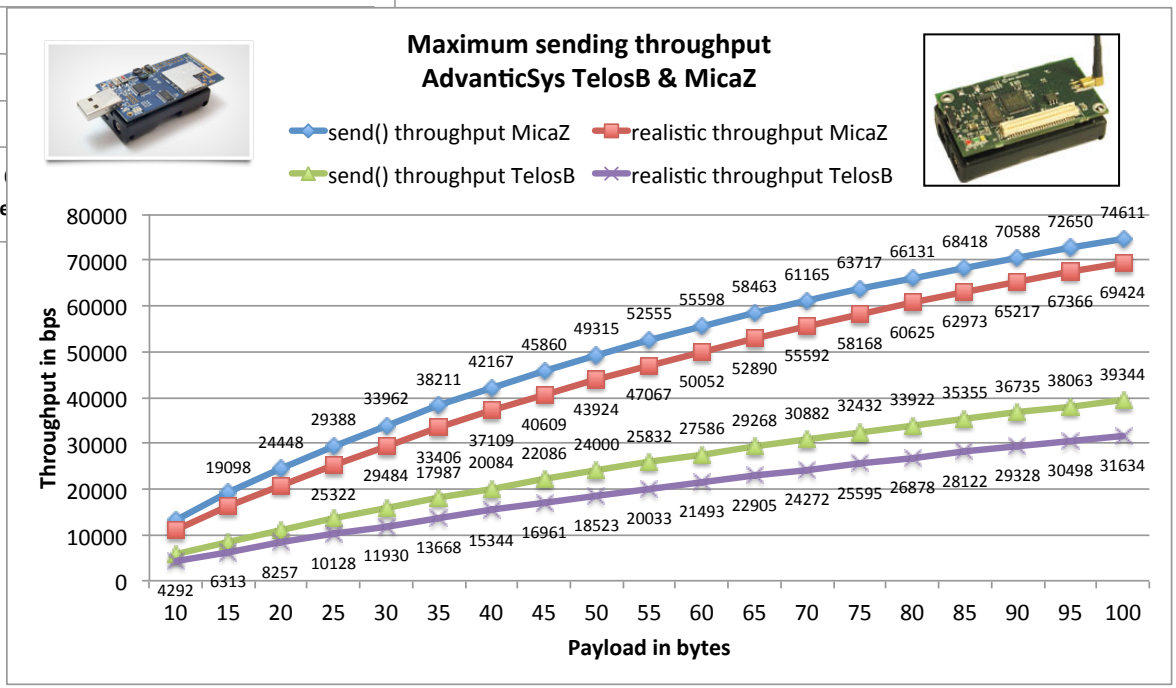


Maximum sending throughput

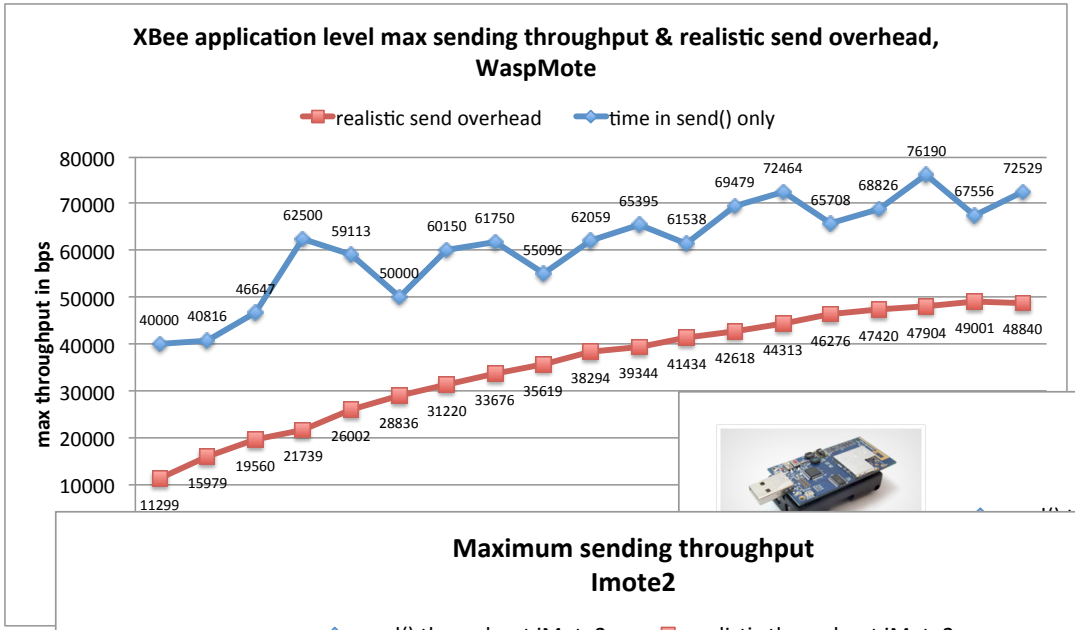


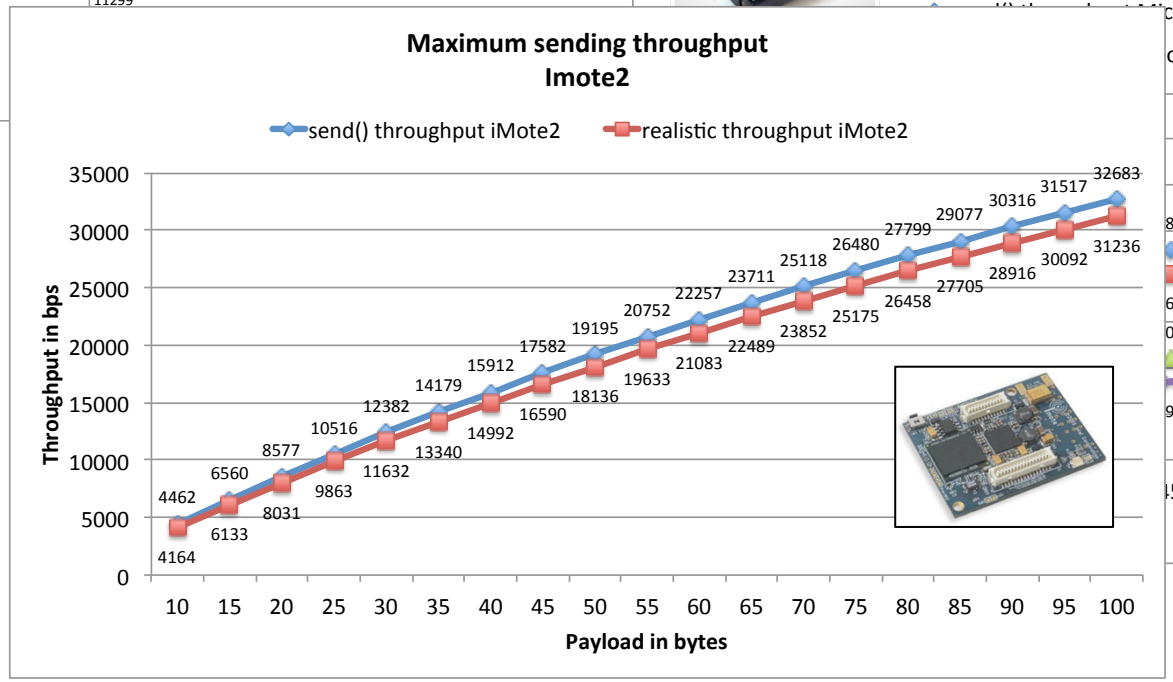
Maximum sending throughput

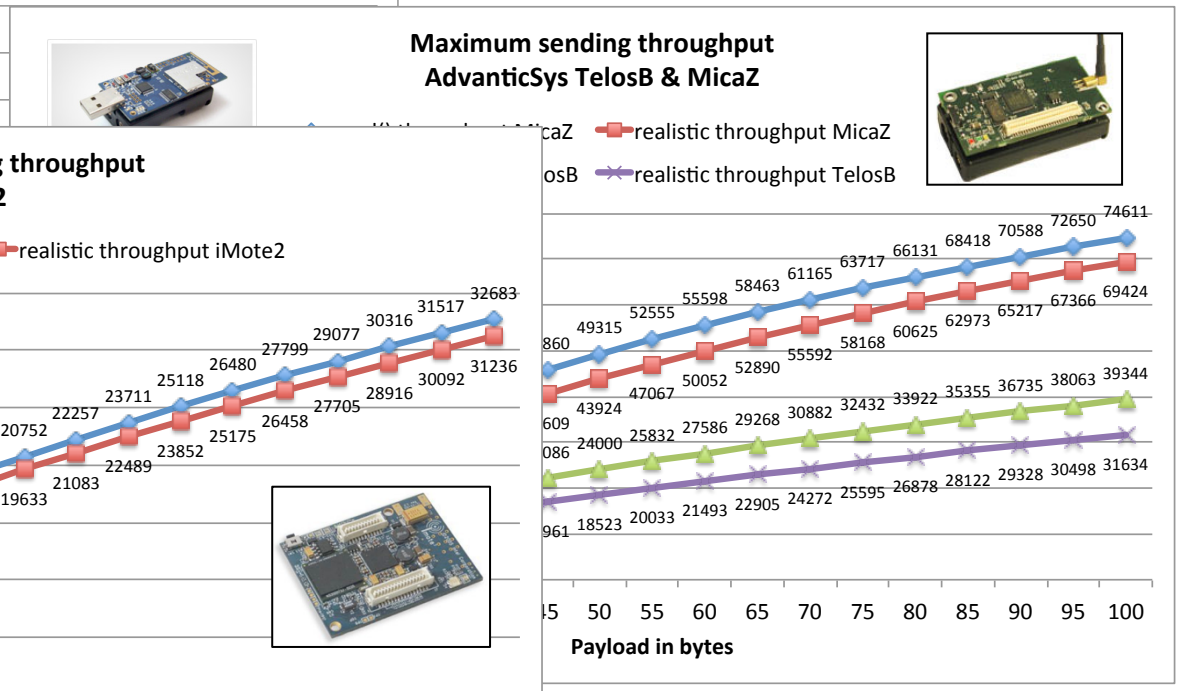




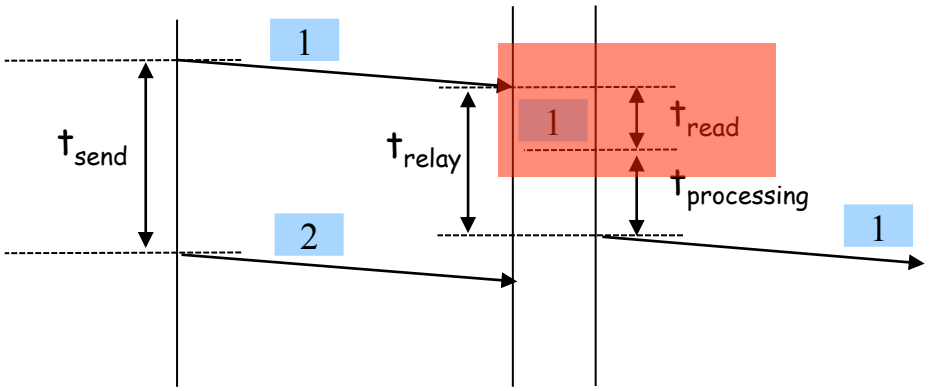
Maximum sending throughput





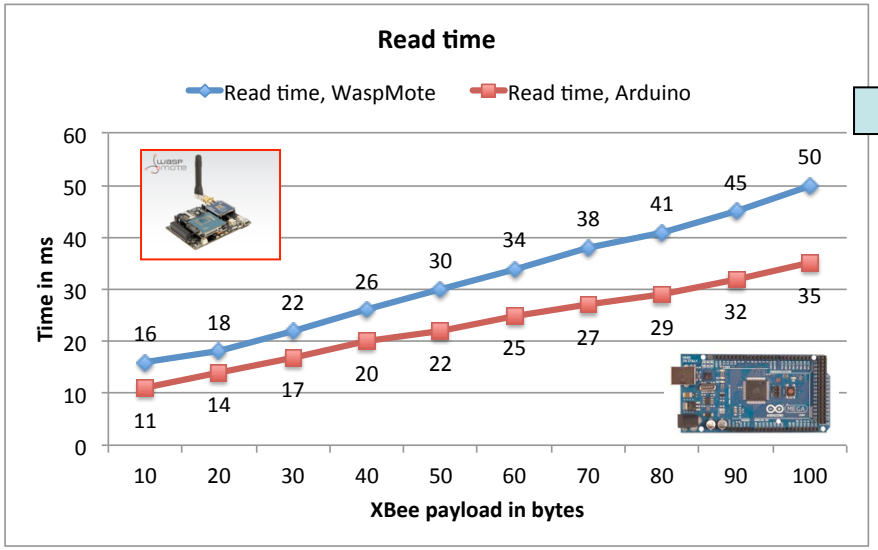


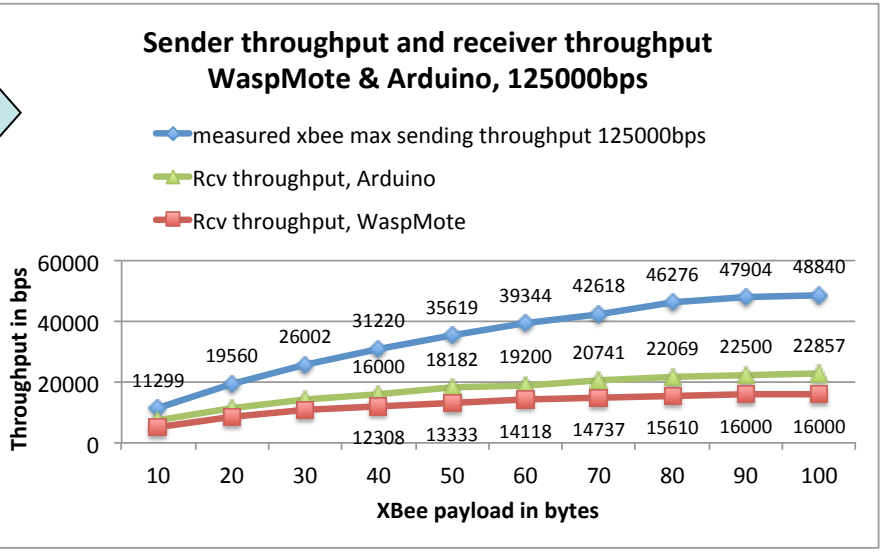
Receive performances

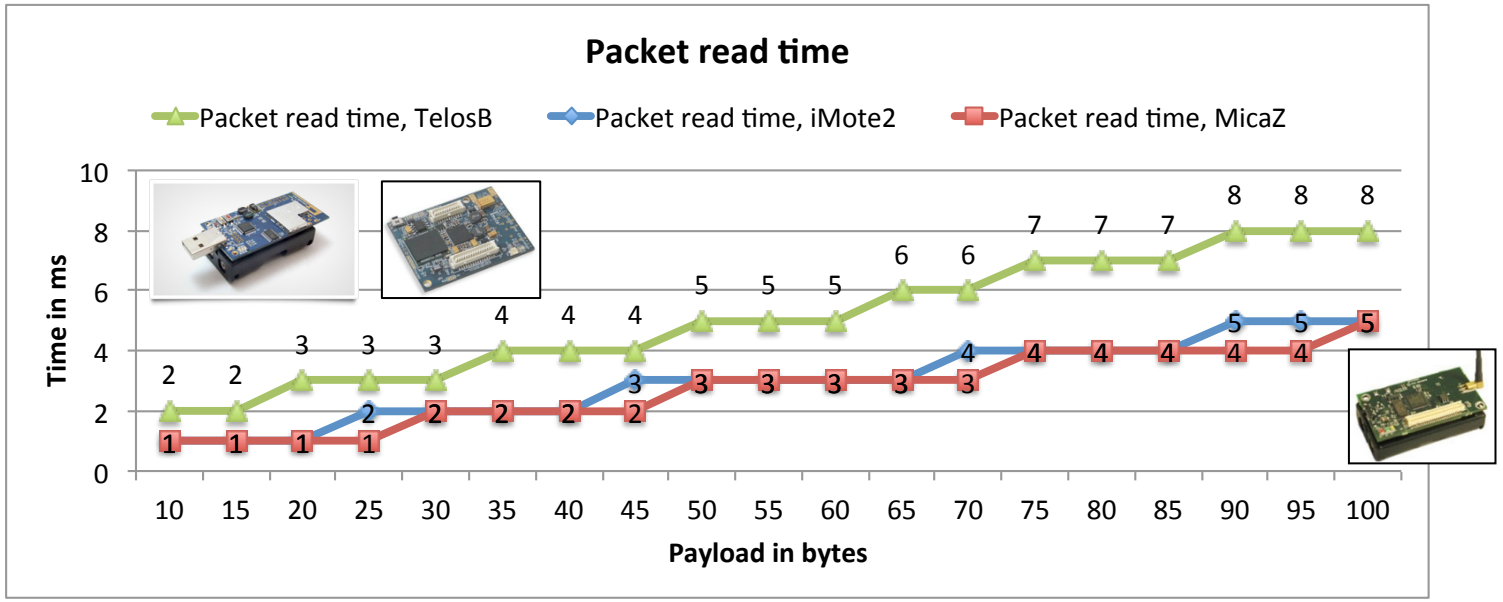


- At sender side, send as fast as possible
- At receiver side, determine T_{READ}
- … And also compute the maximum receive throughput per packet size

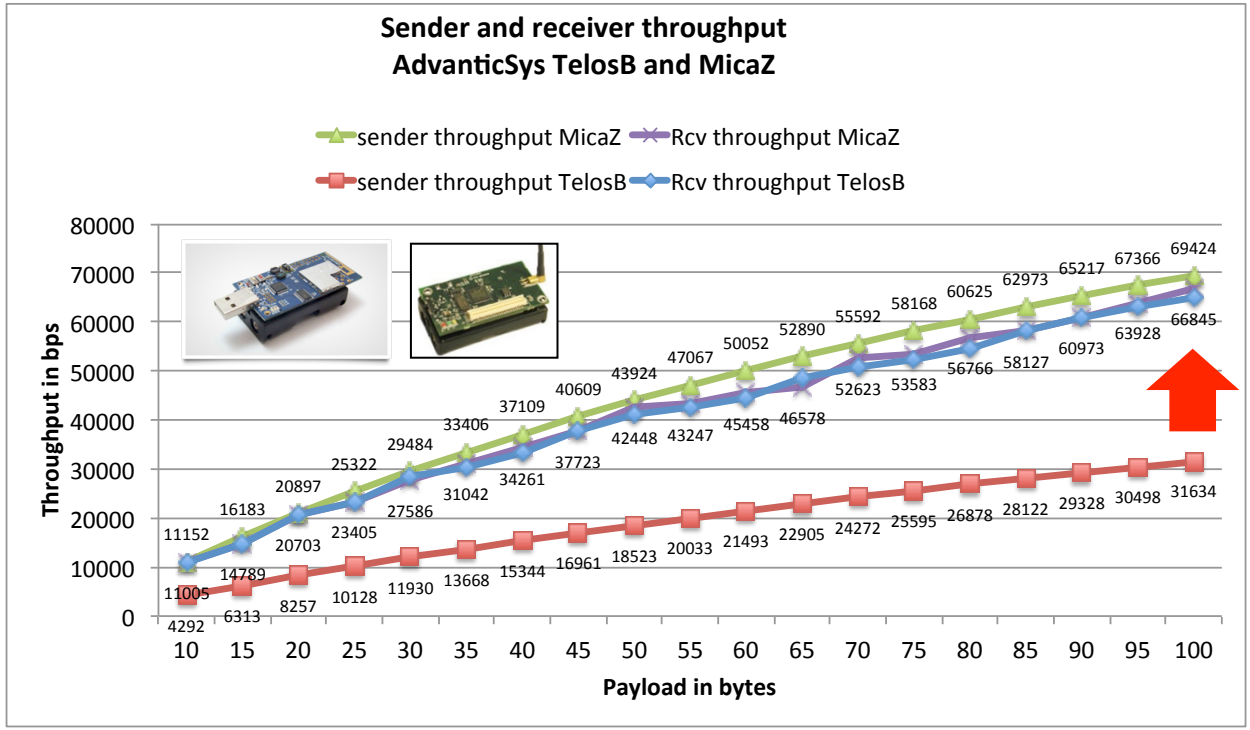
T_{READ} FOR VARIOUS MOTES



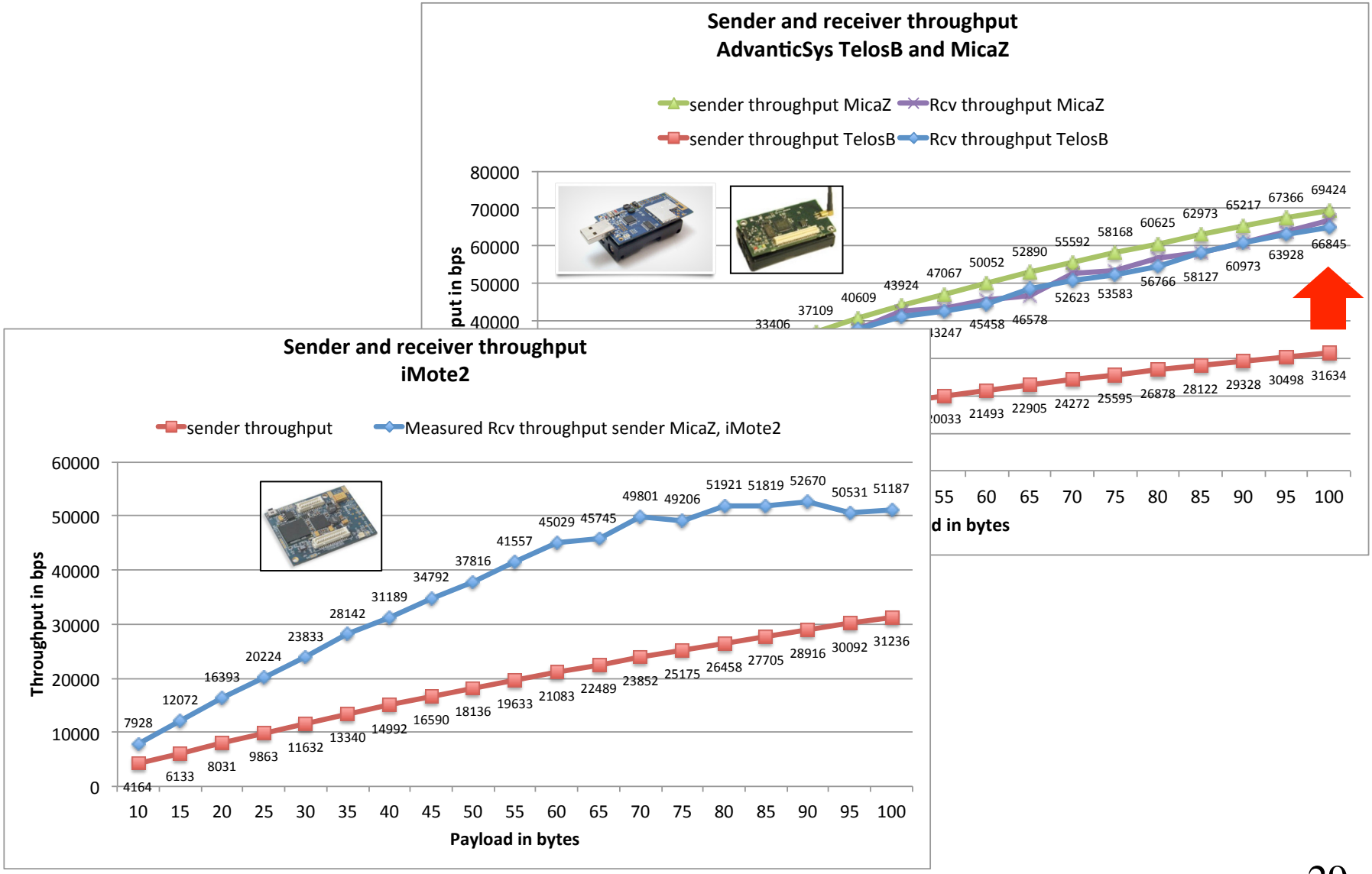




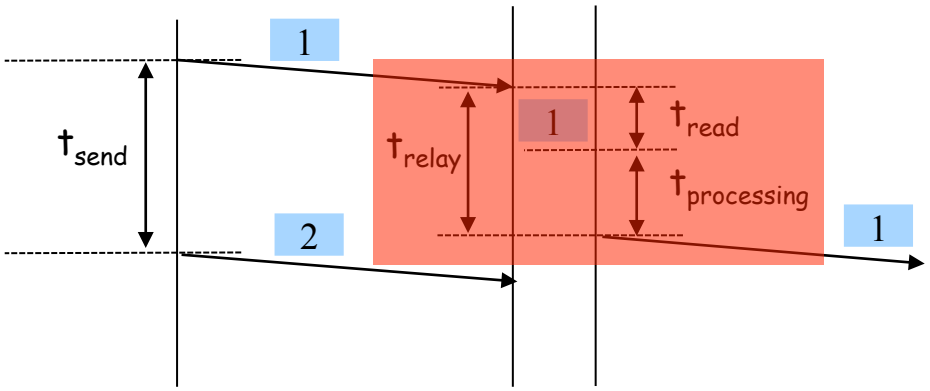
Receiver throughput



Receiver throughput

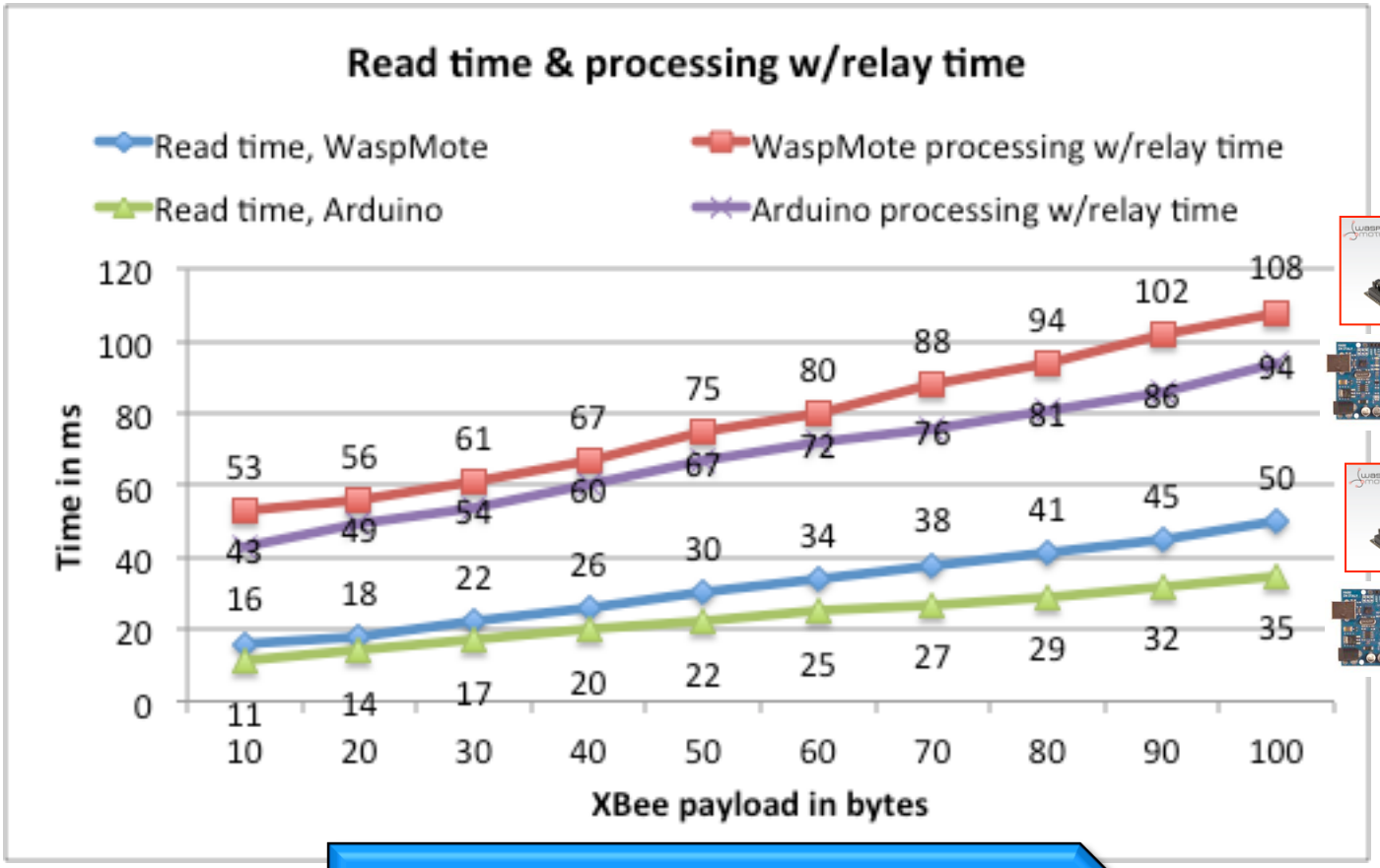


Relay performances



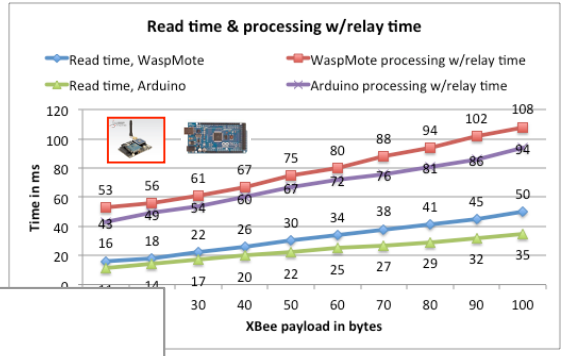
- Relaying are usually done at application-level (even OS level is considered app-level for the mote)
- Relaying means:
  - Read the packet in memory
  - Send the packet to next hop

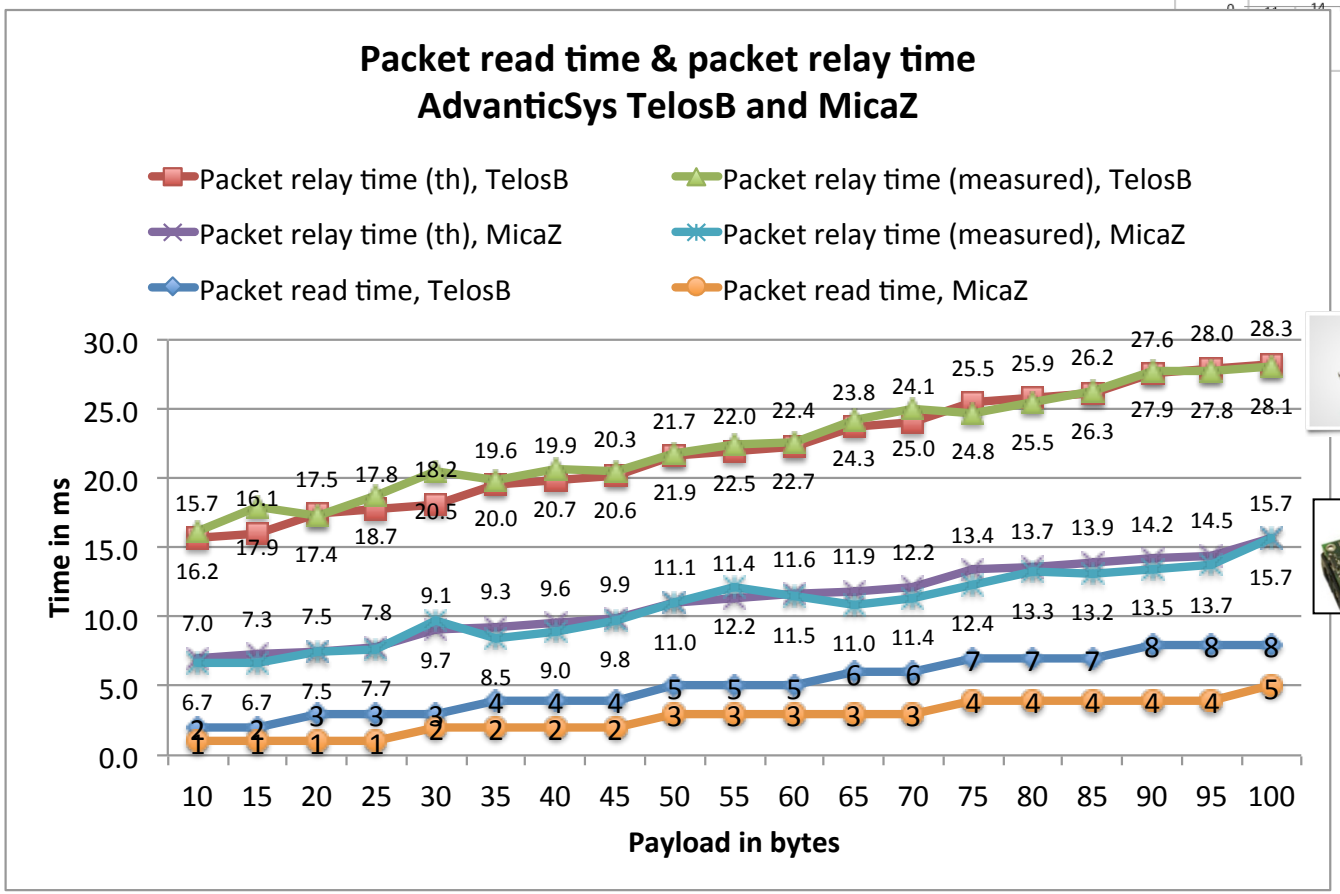
Read time and relay time



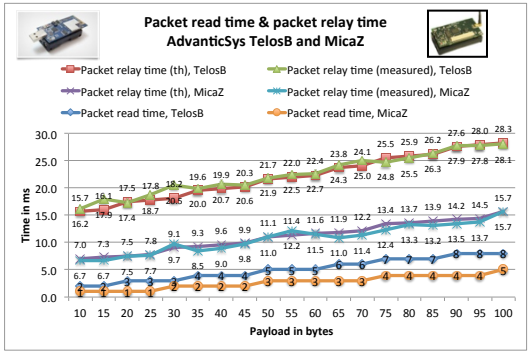
Read time is quite independant from the UART baud rate, but depends on microcontroller frequency

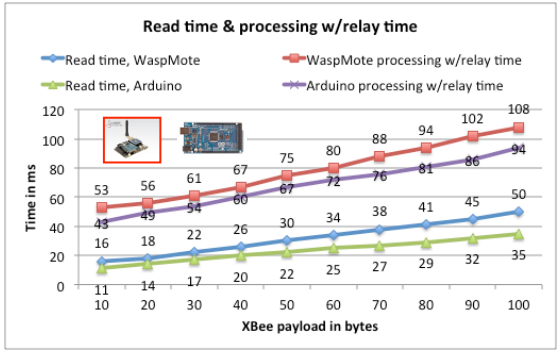
Read time and relay time

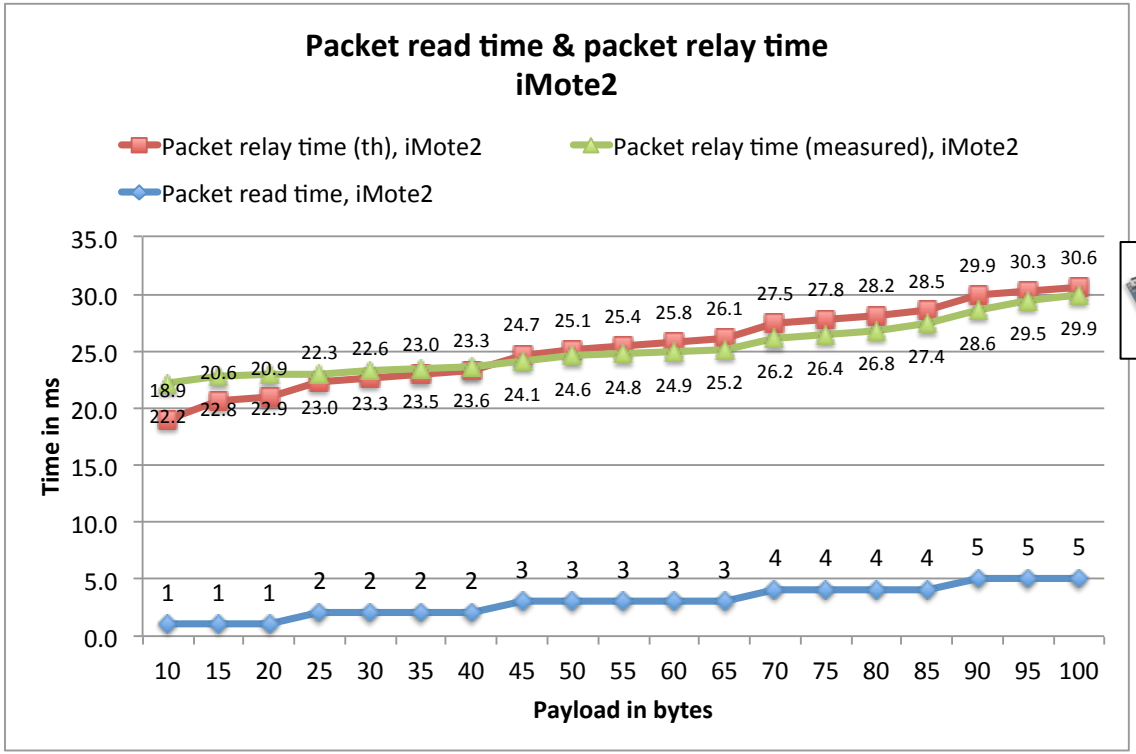




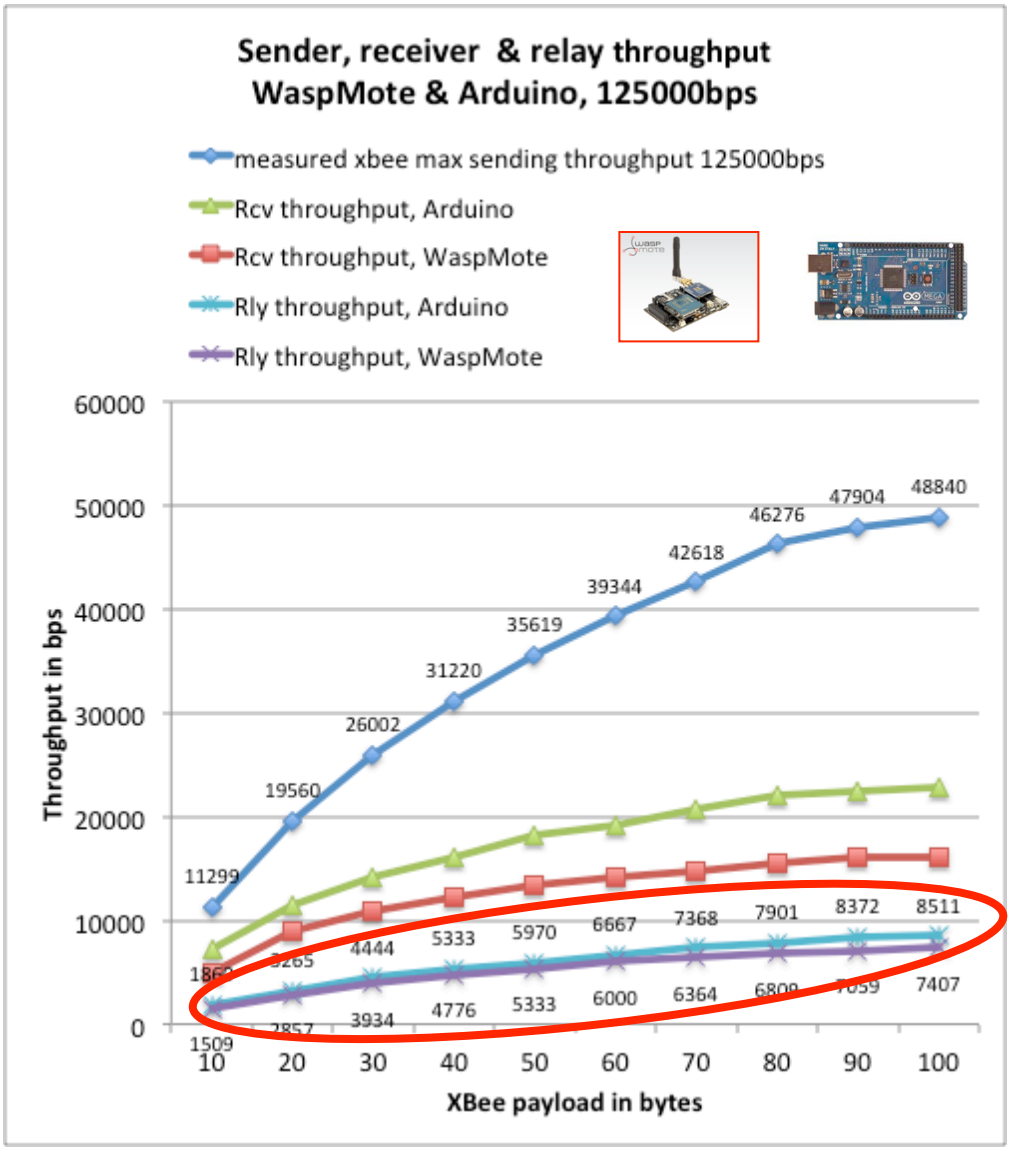
Read time and relay time



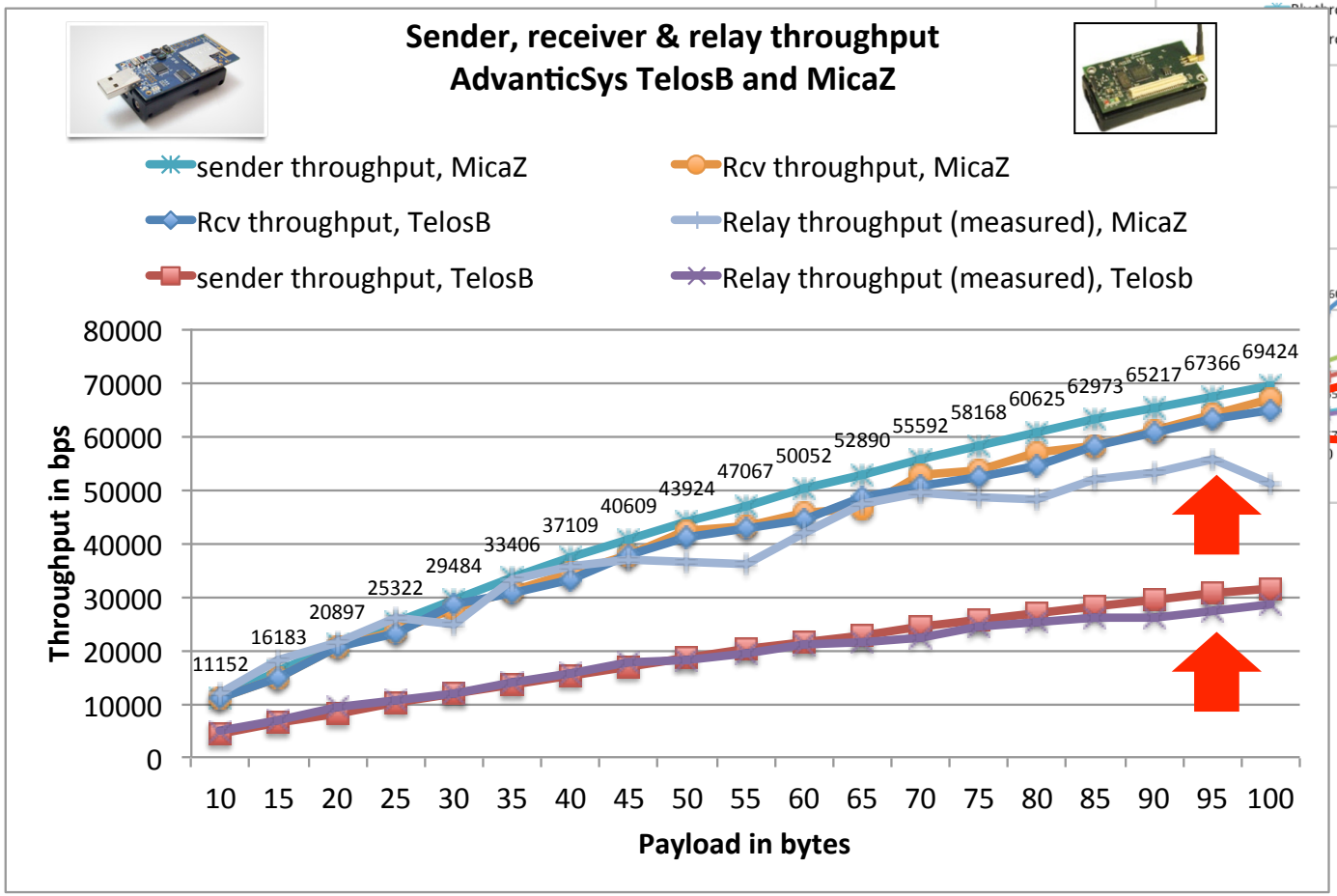


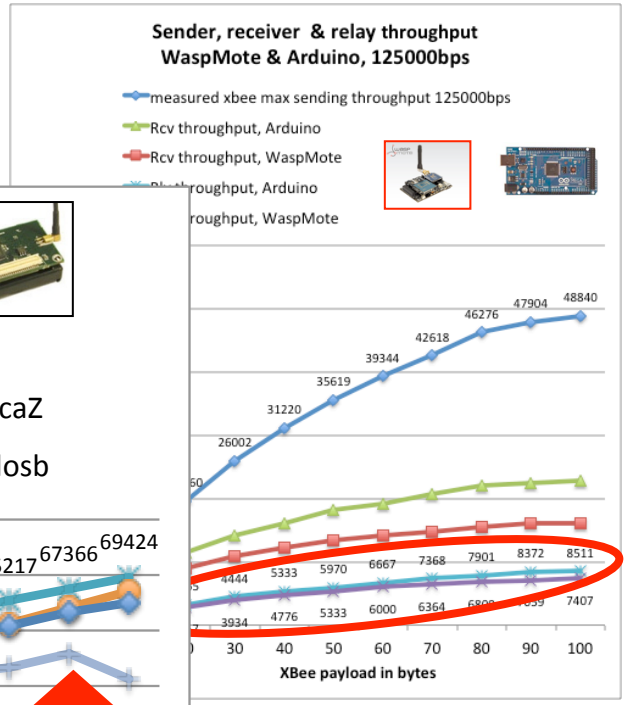


Maximum expected throughput in multi-hop

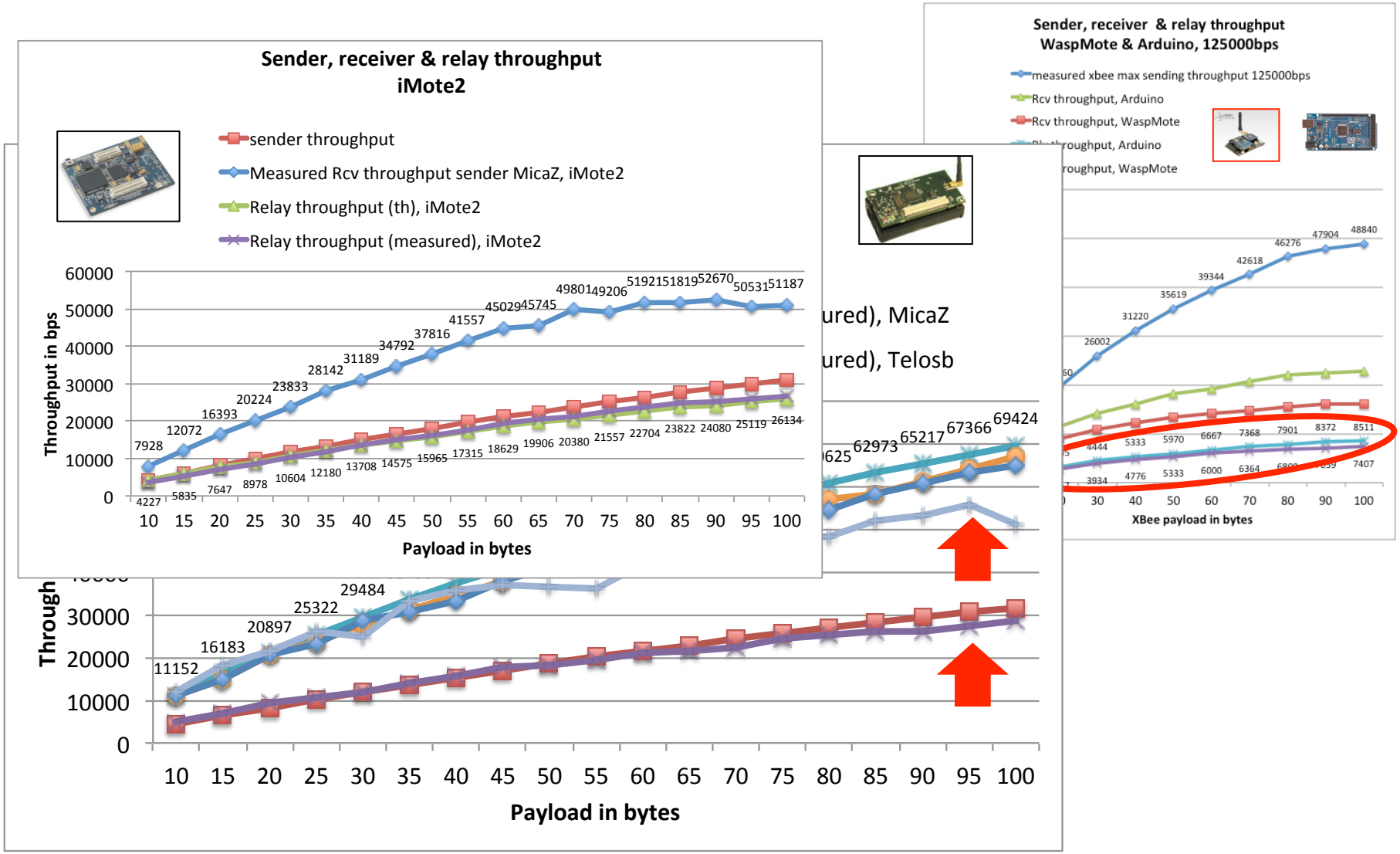


Maximum expected throughput in multi-hop





Maximum expected throughput in multi-hop

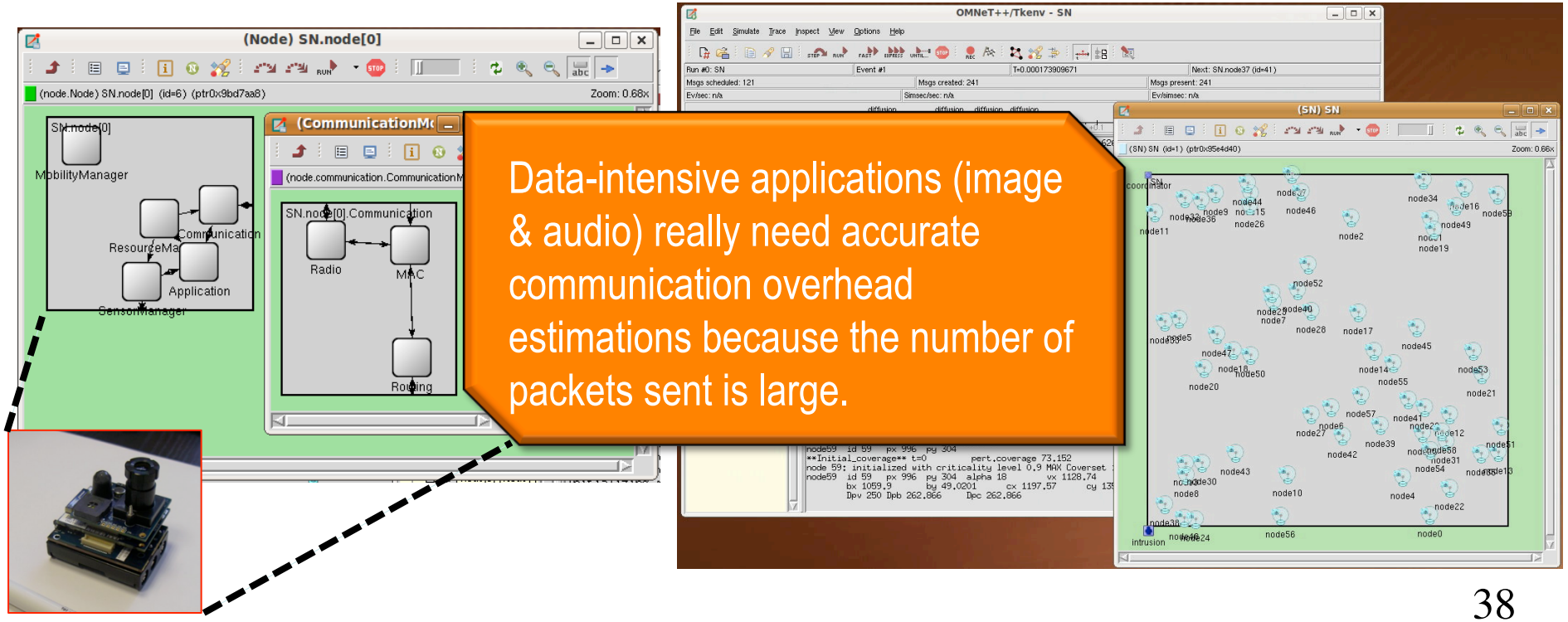


Conclusions

- Looking at app-level performances, taking into account OS&API overheads
- In multi-hop communication, relay time can dramatically reduce the E2E performances
  - Mass-Marcket sensors are quite limited by read overhead
  - MicaZ appears to be the most performant platform
- Can be used to build more realistic simulation model

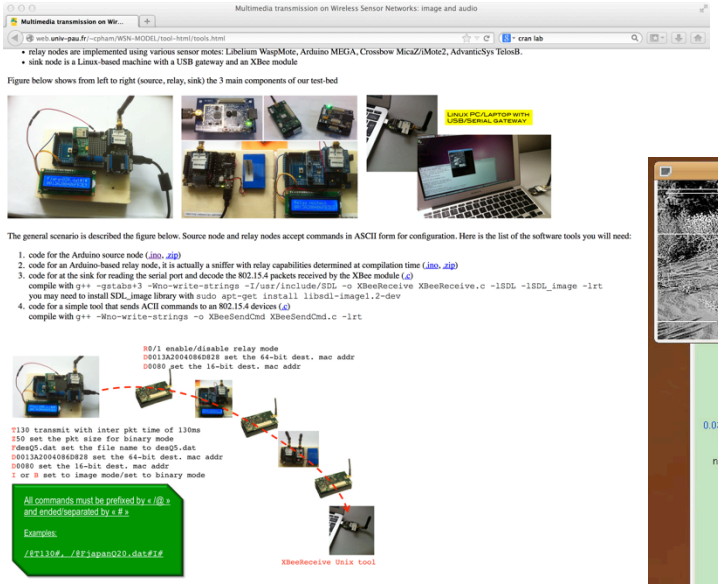
Building more realistic simulation model

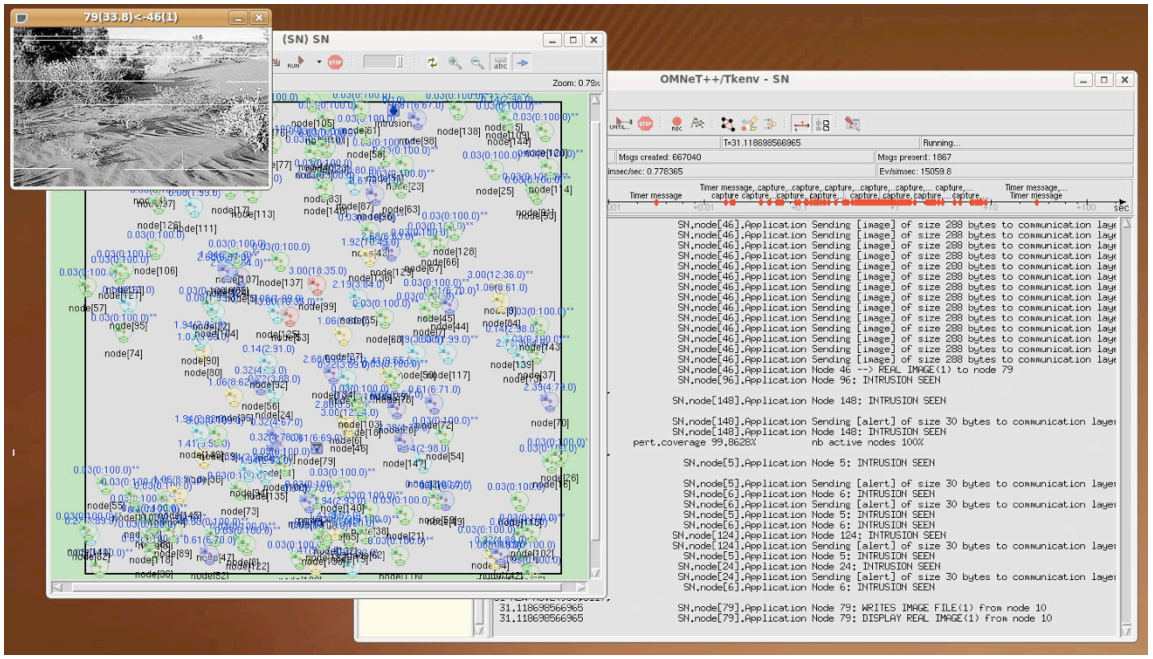
- Don't consider that radio throughput is available at app-level!
- Using only Transmission time is far from being realistic!



Some links

http://web.univ-pau.fr/~cpham/WSN-MODEL/tool-html/tools.html





http://web.univ-pau.fr/~cpham/WSN-MODEL/wvsn-castalia.html