

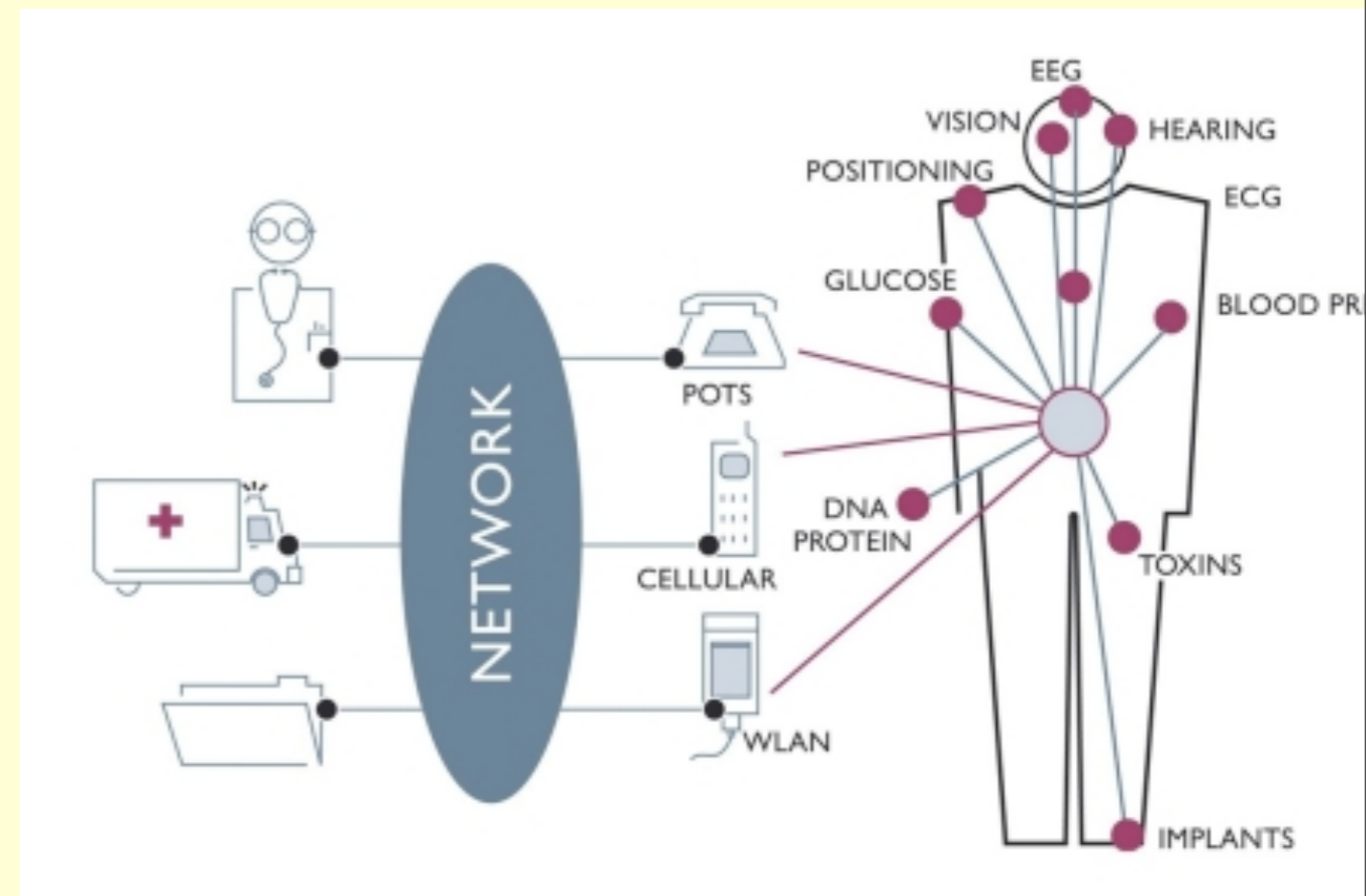
Debugging Sensor Code

Doina Bucur and Marta Kwiatkowska
Computing Laboratory,
University of Oxford



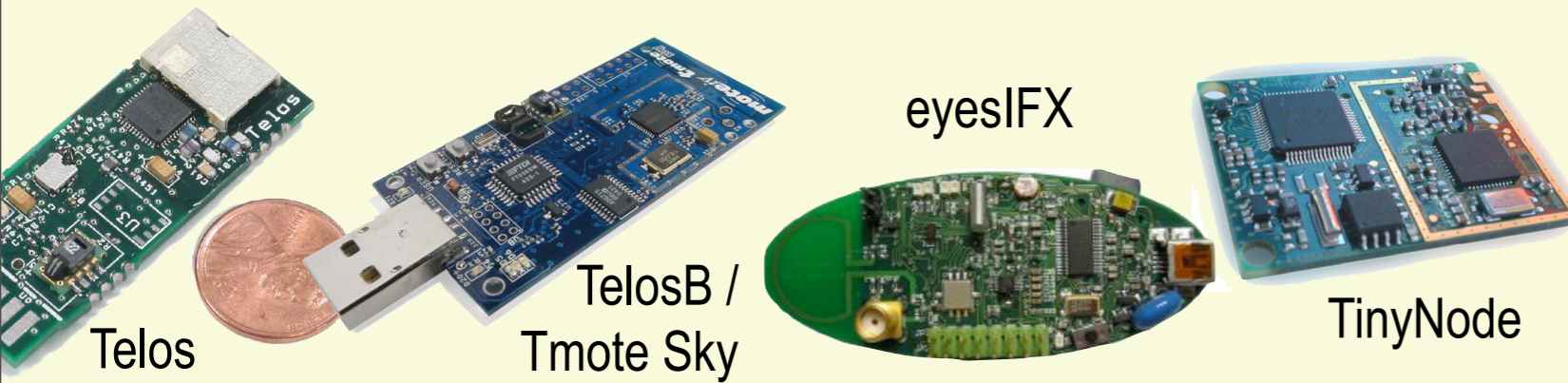
Sensor networks

- Pervasive Healthcare.
Body Sensor Networks
- TinyOS:
 - Modern OS and language in an embedded system

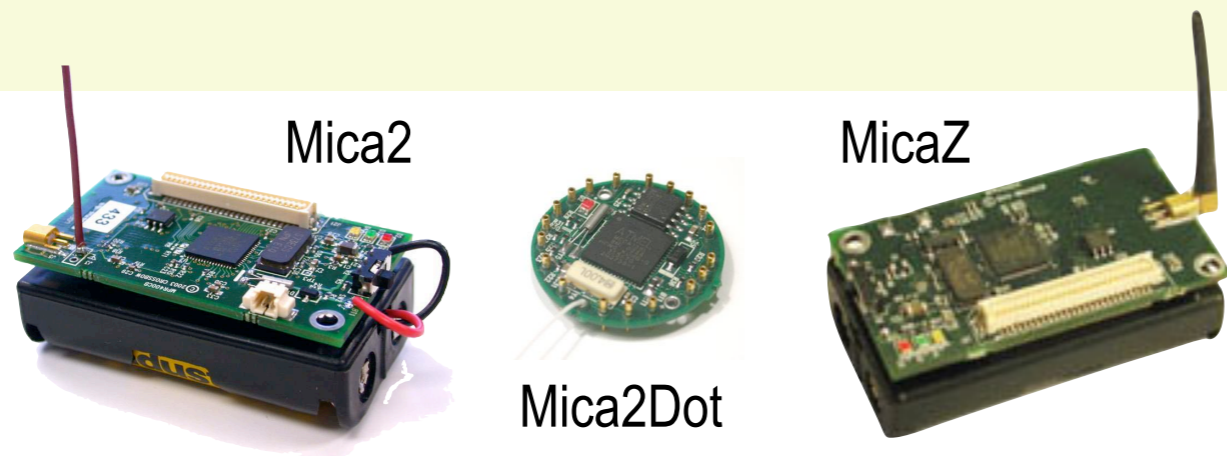


Architectures

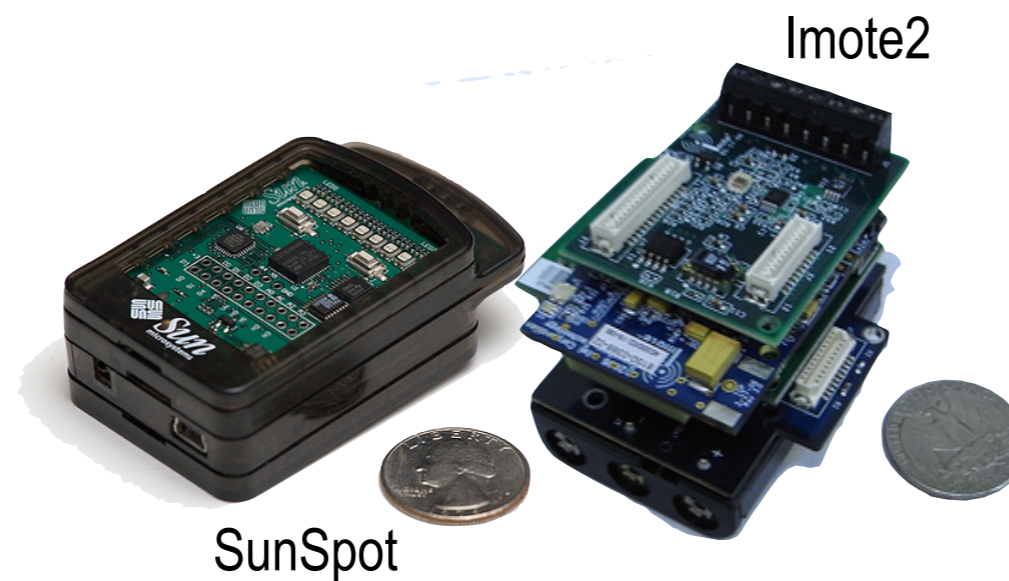
MCU



MSP430
(Texas Instruments)



AVR
(Atmel)



ARM

This talk is about

trustworthy sensor software

on the **dark side** of Aml (see keynote).

- **Policies** upon software
 - Regulate software deployed on body sensor networks, as laws regulate a doctor's behaviour

Software

--- **Bugs**: The horror stories



European Space Agency
Ariane 5
first launch, 1996

- Legacy **Ariane 4** code
- Failure: velocity encoded overflowed 16 bits
- Autodestruct



Atomic Energy of Canada
Therac-25
radiation therapy machine
1985-87

- Patients given deadly radiation overdoses (x100)
- Race condition



Toyota Prius 2004-2005

- Hybrid engine
- Gas engine would stall at high speeds

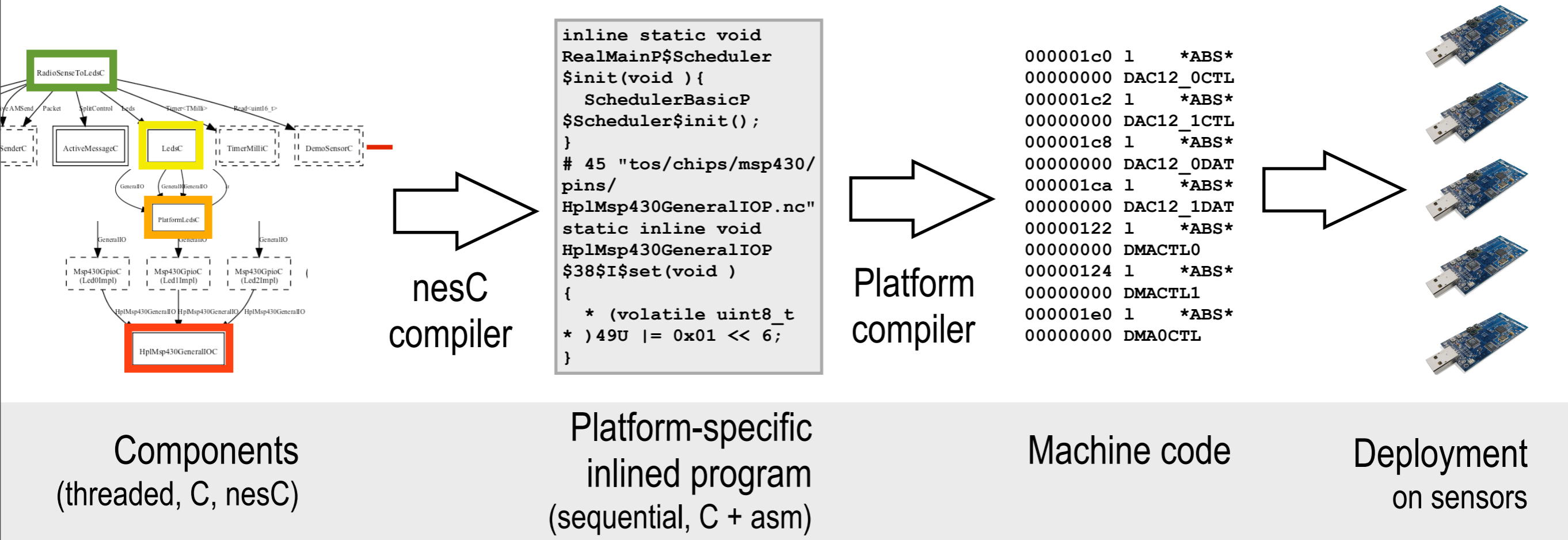
Bugs: The horror stories

Software bugs cost an economy about 0.6% of GDP
[US, 2002]

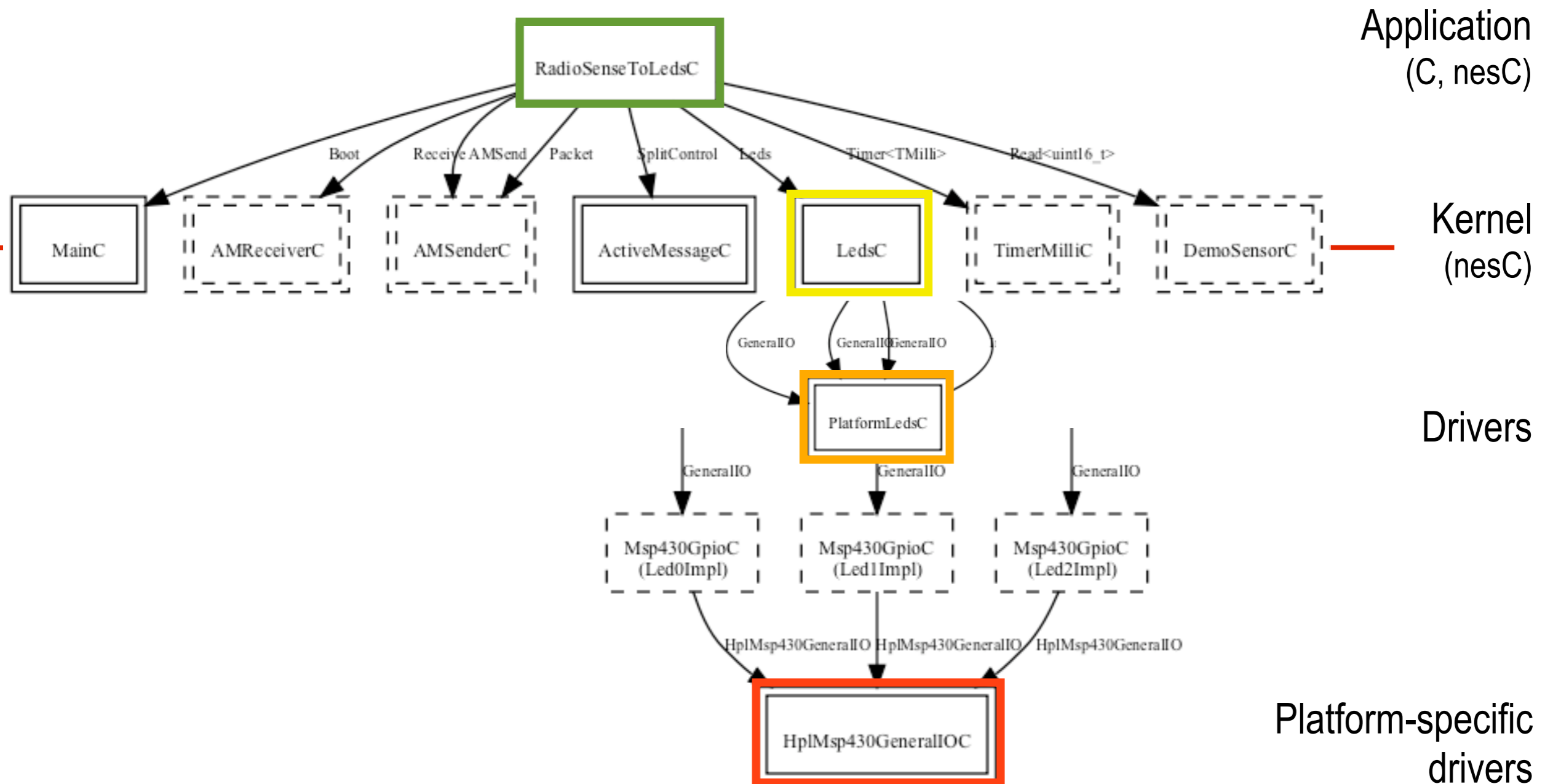


Subtle
Crash the program
Security and human issues

TinyOS tool chain

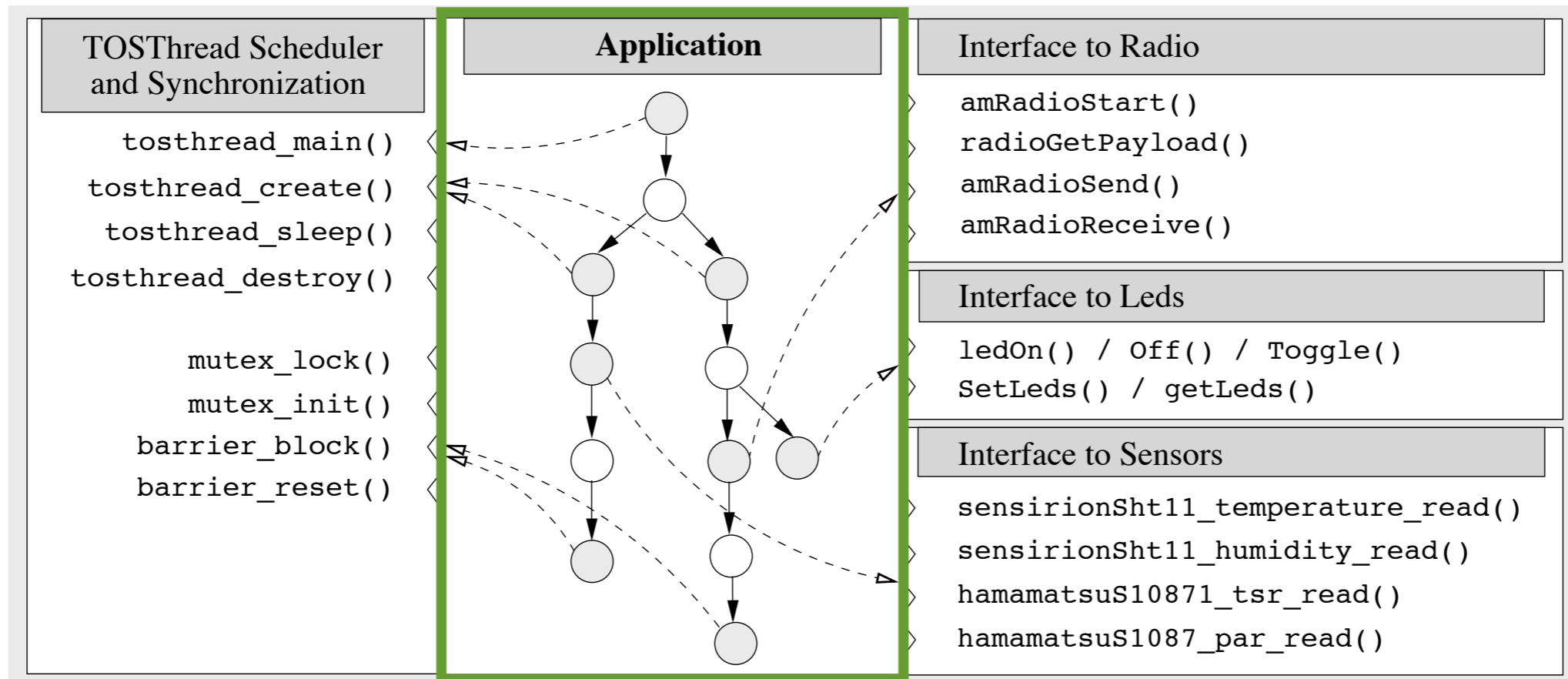
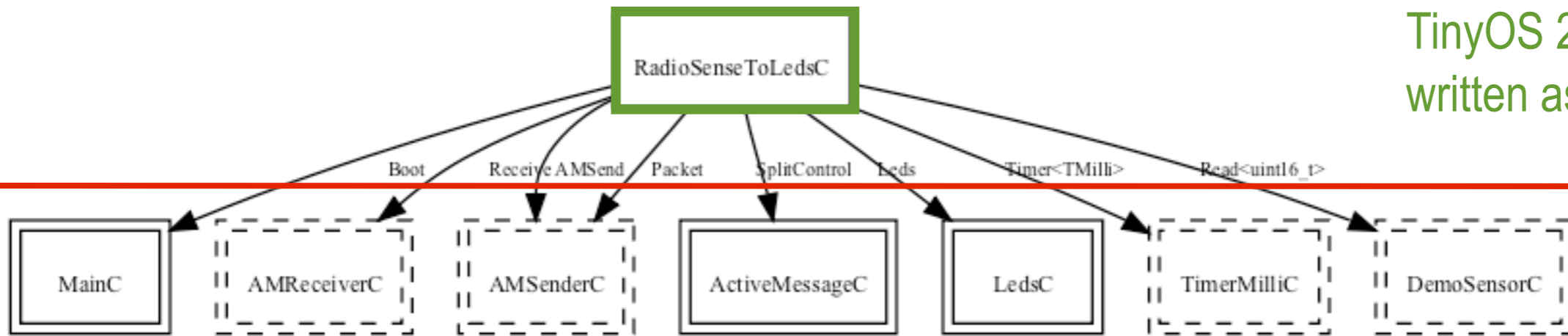


A TinyOS application



Application model

TinyOS 2.x applications written as TosThreads.



Software verification

- **Verification** vs. simulation.
- Counterexample-guided abstraction refinement (**CEGAR**)
- **CProver** tools [cprover.org].

Bugs (Policies)

Classical case:
context-aware code not coping with **contextual exceptions**

Categories of bugs in context-aware, TinyOS applications

Sensing exceptions Incomplete treatment of sensing errors.

Network exceptions Incomplete treatment of network errors.

Interface use Incorrect use of interface to kernel services.

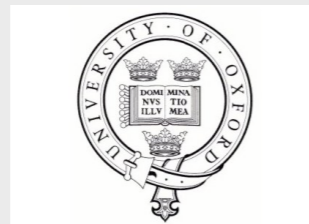
False reasoning Incorrect decision-making given a context situation.

Tool run times

Application (Threads/LOC)	Claim line	Verified?	Time	Bug: context awareness
<i>Blink</i> 4/64	66	yes	2.9s	-
<i>SenseAndSend</i> 6/347	79	no	32.2s	interface use
	136	no	1m08s	sensing exception
	146	yes	4m25s	-
<i>PatientNode</i> 6/439	172	yes	29.9s	(interface use)
	254	yes	3m55s	(sensing exception)
	230	no	35m07s	network exception
	268	yes	2m38s	(false reasoning)
	262	yes	61m12s	(false reasoning)

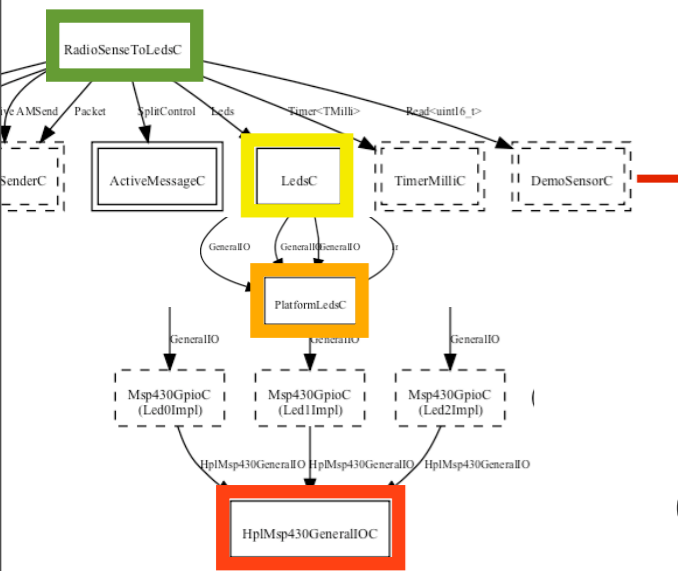
Debugging Sensor Code

Doina Bucur and Marta Kwiatkowska
Computing Laboratory,
University of Oxford



Thank You!

Current work



nesC
compiler

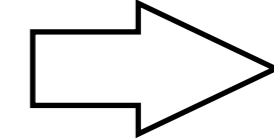
```

inline static void
RealMainP$Scheduler
$init(void ){
    SchedulerBasicP
    $Scheduler$init();
}
# 45 "tos/chips/msp430/
pins/
HplMsp430GeneralIOP.nc"
static inline void
HplMsp430GeneralIOP
$38$I$set(void )
{
    * (volatile uint8_t
    * )49U |= 0x01 << 6;
}
    
```

Platform
compiler

```

000001c0 1    *ABS*
00000000 DAC12_0CTL
000001c2 1    *ABS*
00000000 DAC12_1CTL
000001c8 1    *ABS*
00000000 DAC12_0DAT
000001ca 1    *ABS*
00000000 DAC12_1DAT
00000122 1    *ABS*
00000000 DMACTL0
00000124 1    *ABS*
00000000 DMACTL1
000001e0 1    *ABS*
00000000 DMAOCTL
    
```



Components
(threaded, C, nesC)

Platform-specific
inlined program
(sequential, C + asm)

Machine code

Deployment
on sensors

