python-csp
CSP as a DSL for Python and Jython

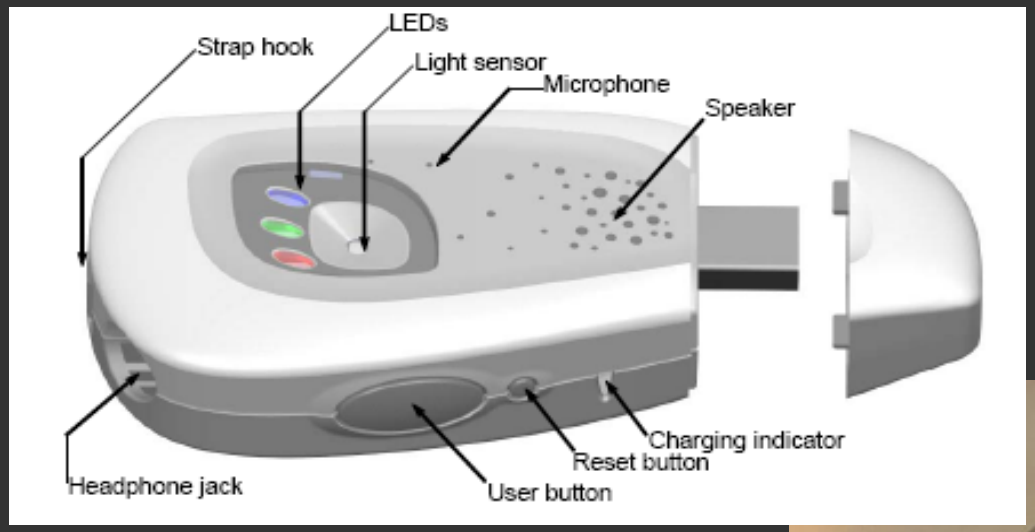
Sarah Mount, University of Wolverhampton

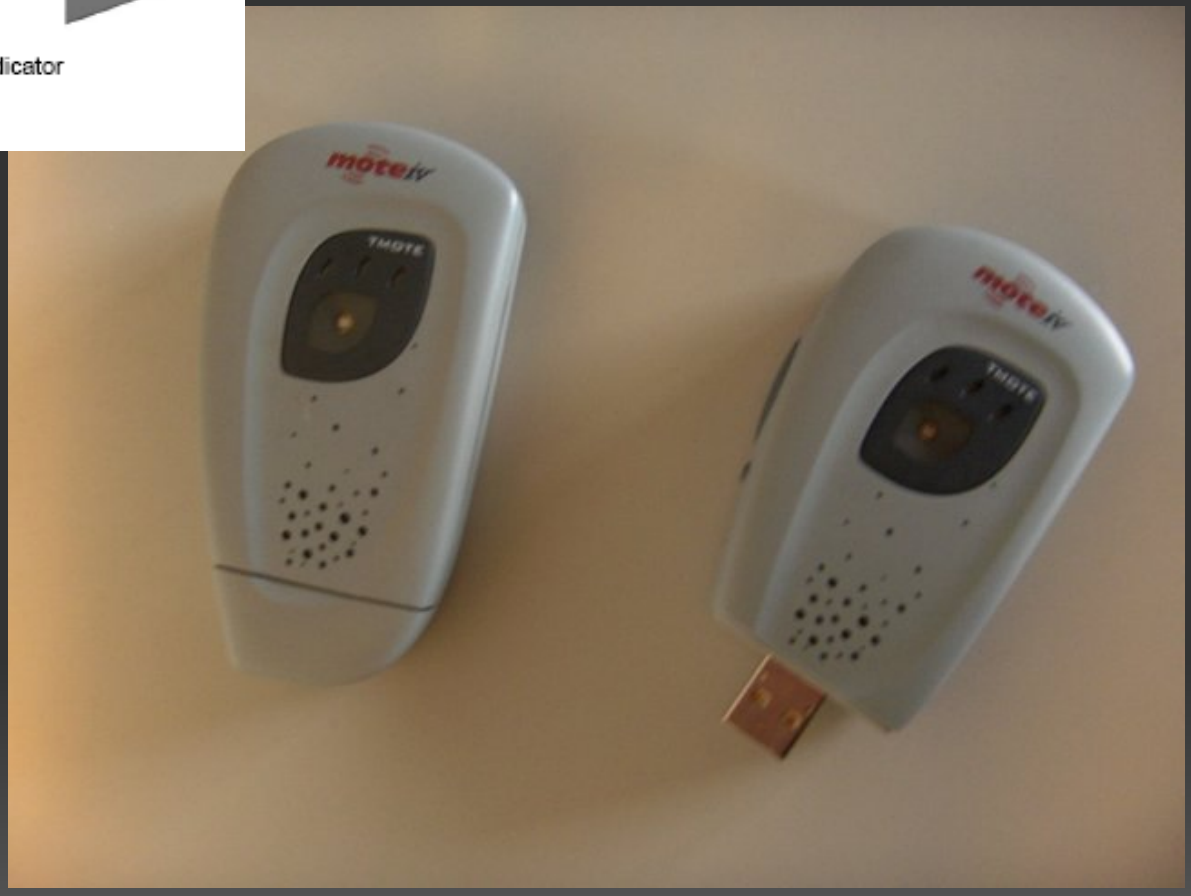
w:	http://www.snim2.org
e:	s.mount@wlv.ac.uk
t:	@snim2

Contents...

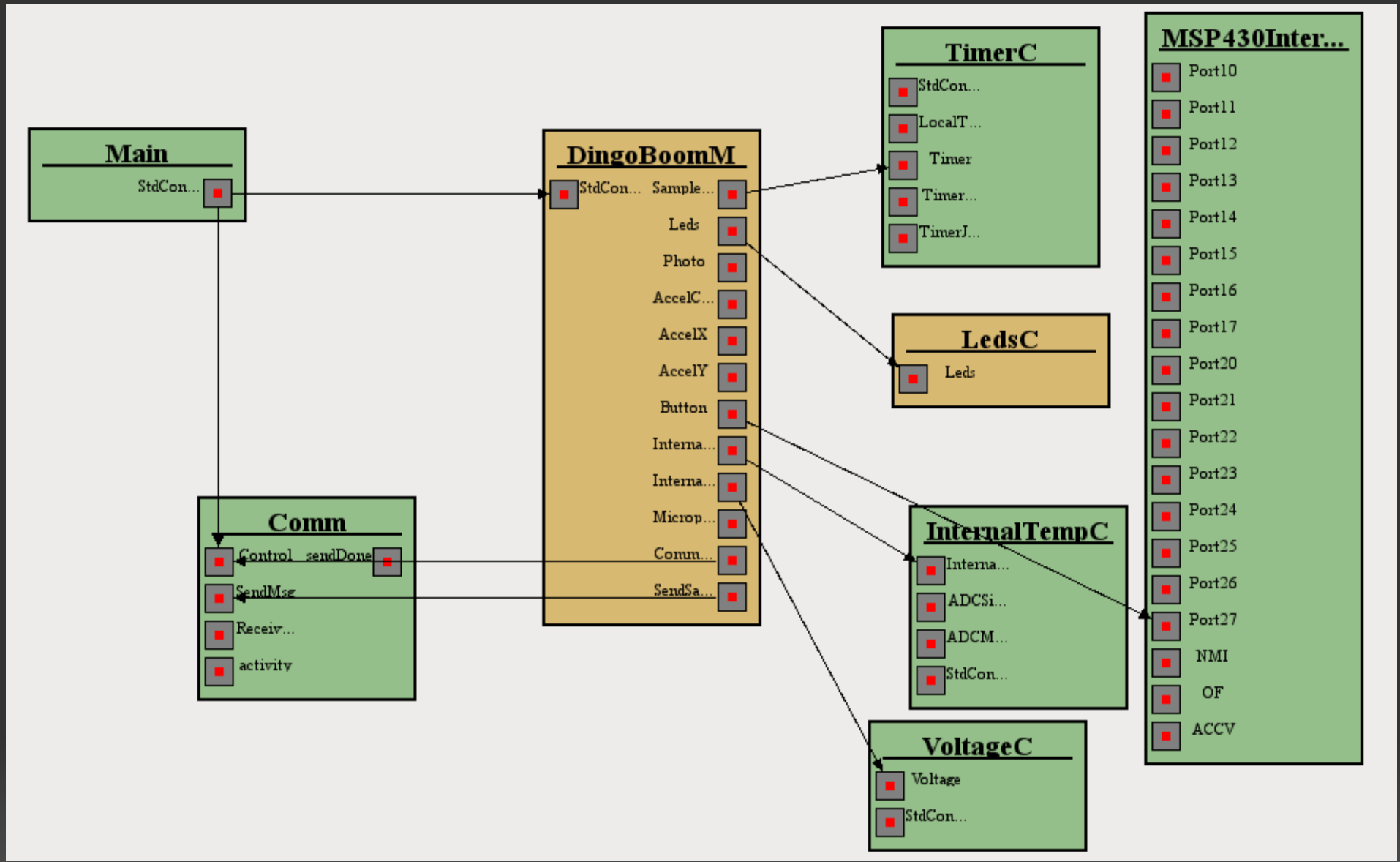
- Why we did this
- Advantages / disadvantages of Python for highly concurrent or process oriented work
- General theme of python-csp
- Syntax / semantics / examples
- Future directions

The story of this work ...





Tmote Invent platform from MoteIV (now Sentilla)

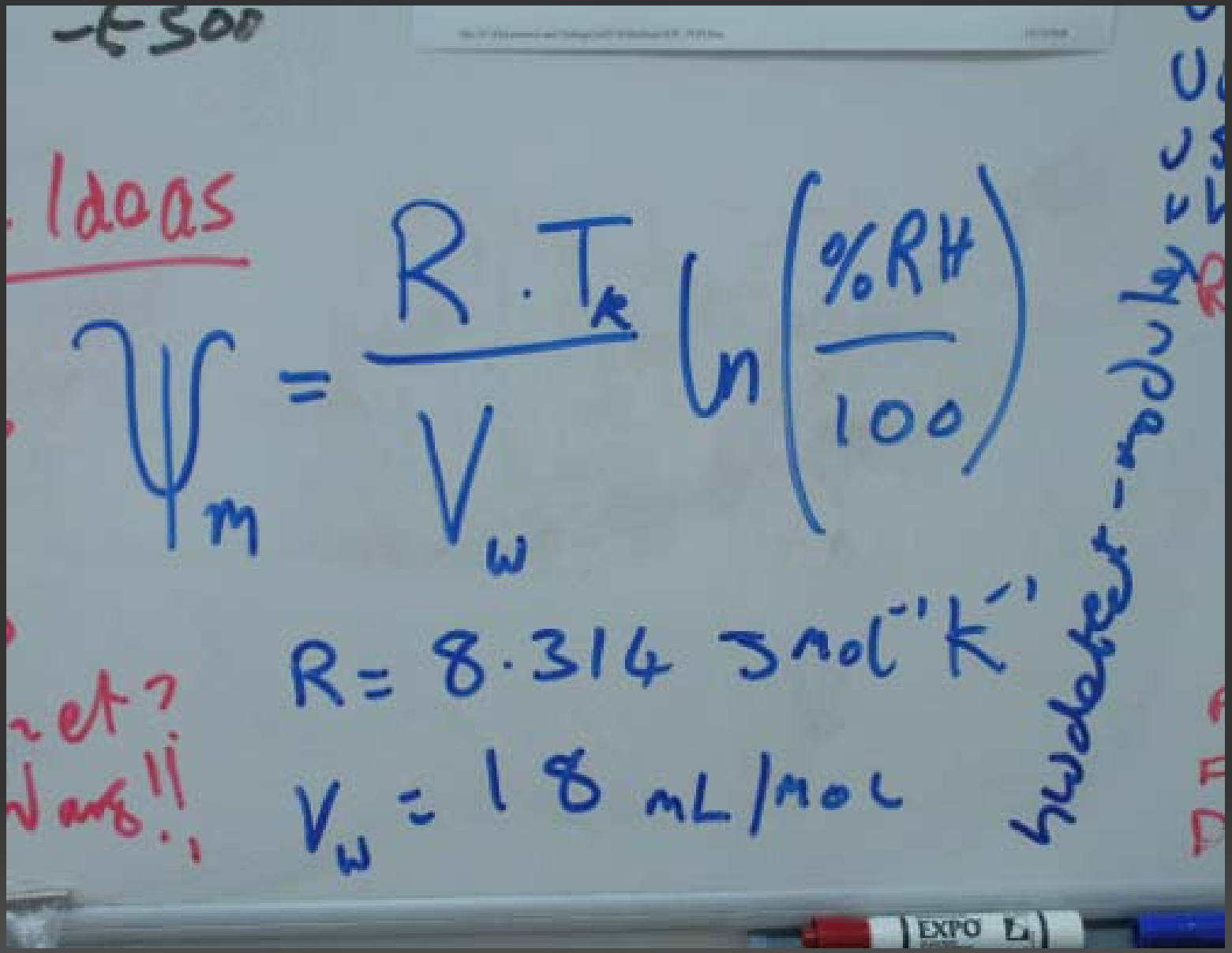


TinyOS code to gather raw data from Tmote Invents

```
module HL2ControllerM
{
  provides interface StdControl;
  uses { ... interface ADC as AccelX; ... }
}
implementation {
  task void getAccelXData() {
    call AccelX.getData();
  }
  async event result_t AccelX.dataReady(uint16_t
                                        data) {
    atomic am->accelX[nextX++] = data;
    post getAccelYData();
    return SUCCESS;
  }
}
```

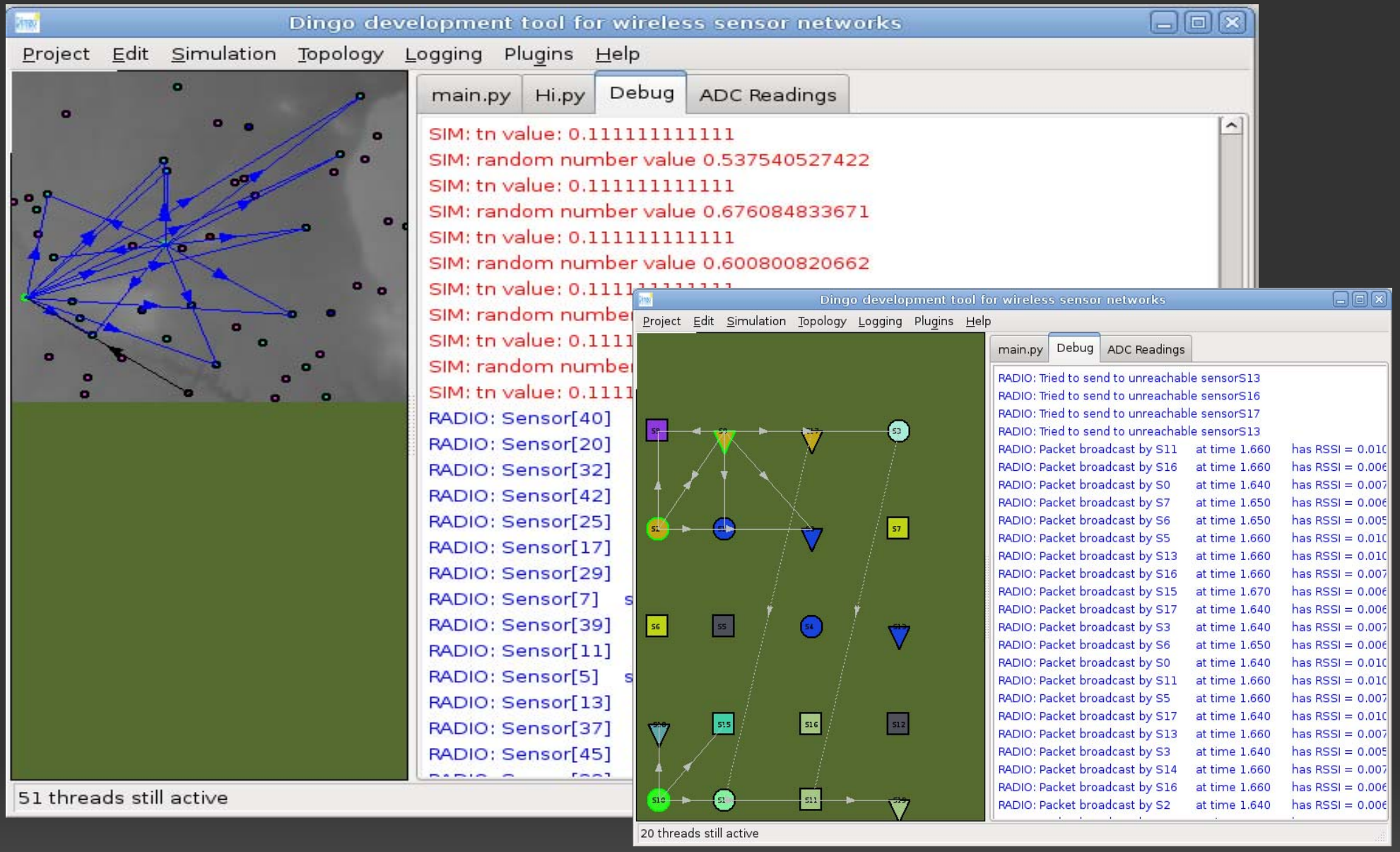
... but what about the application layer?

Soil science and agronomy

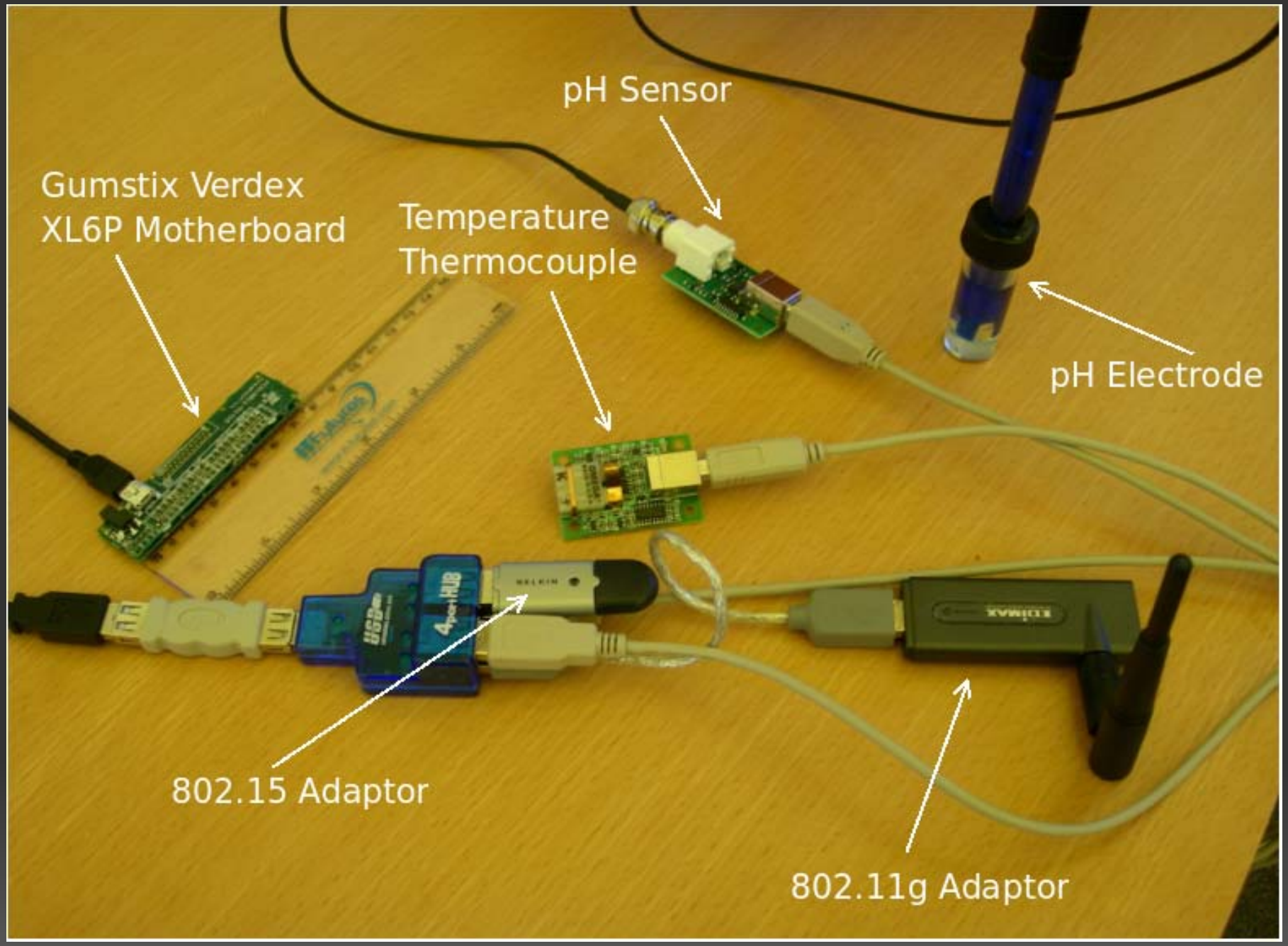


%RH to Soil Matric Pressure

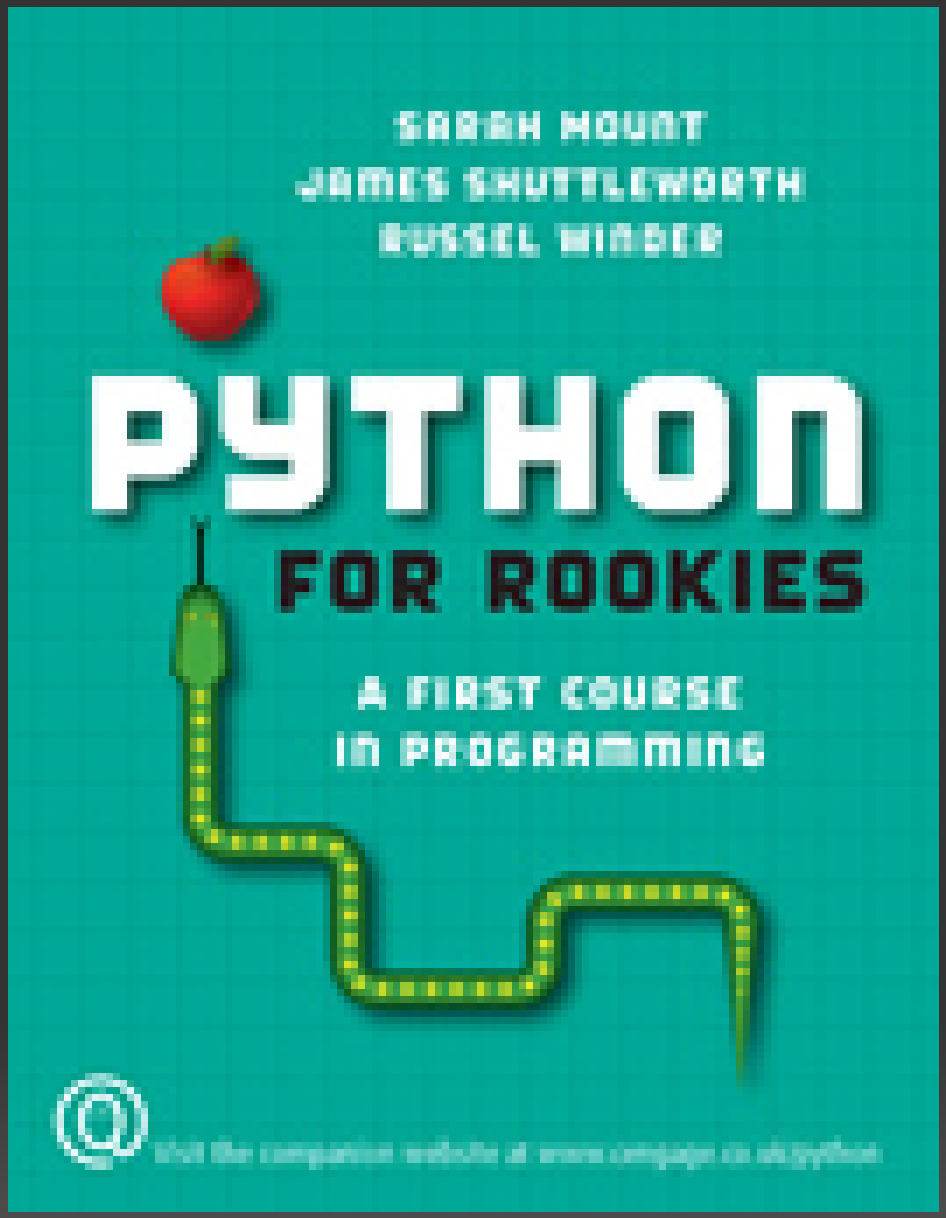

GOODYEAR



Then I made SenSor and Dan Goldmsmith made SensorPlus



Laboratory hardware running Dingo



... so we wrote a book about it all ...

Why python-csp

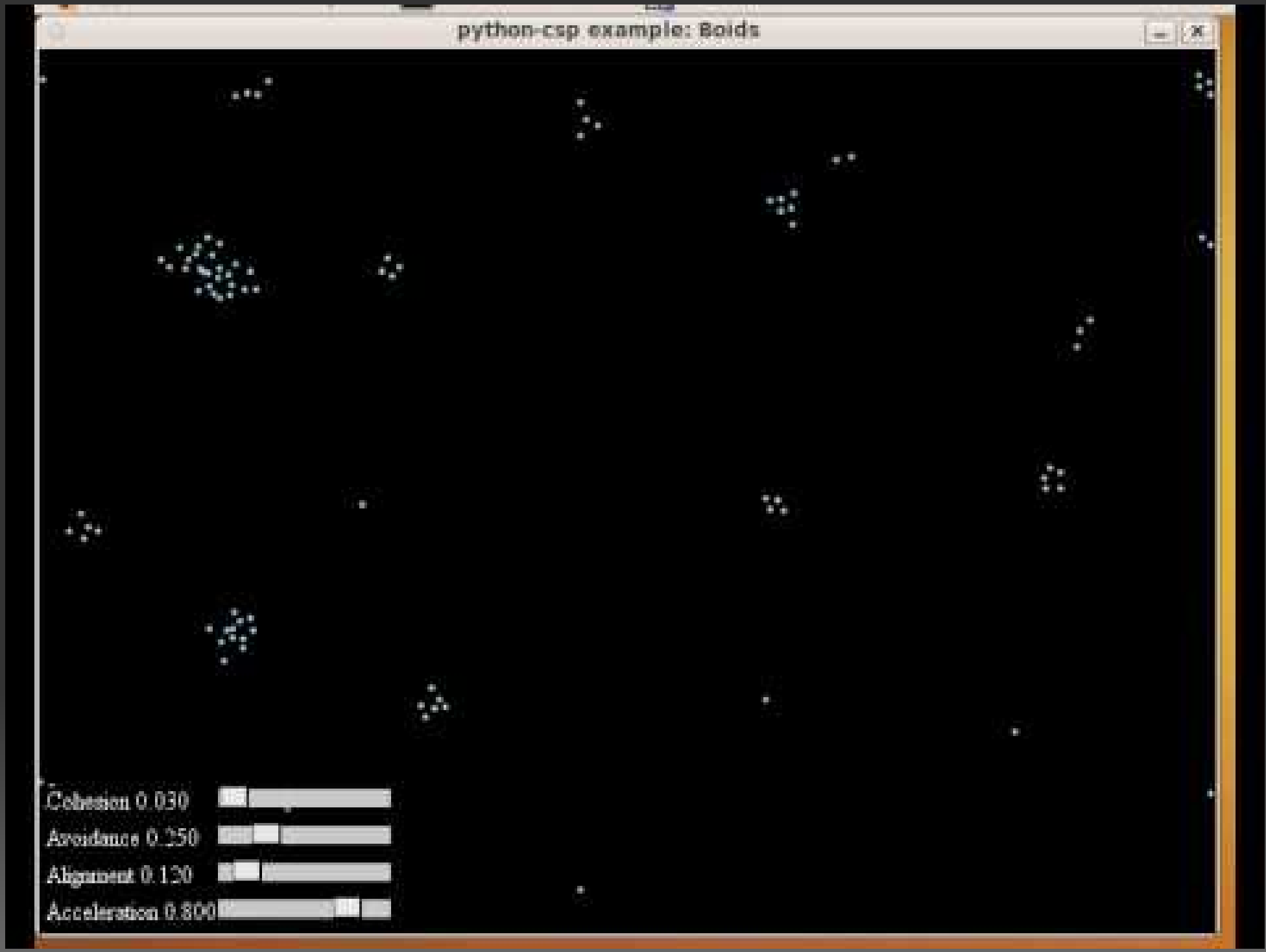
- Keep all the increased productivity and fun of Python
- Add scalable, mobile concurrency
- Profit.

Commstime results

	Mean (micro s)	s.d.
JCSP (Java threads)	23.8	4.29
PyCSP (Processes)	394.97	75.82
PyCSP (Threads)	292.2	47.21
PyCSP (Greenlets)	24.41	0.36
python-csp (Processes)	116.75	35.53
python-csp (Threads)	225.77	17.51
jython-csp (Java threads)	157.8	30.78

Python oddities

- The Beazley effect
  - A multi-threaded algorithm can be slower than a single-threaded algorithm
  - GIL preempts every *$X* OPCODES
- The state of Python's low-level threading libraries
  - Implement POSIX threads
  - Locking facilities (condition variables, locks, mutexes, semaphores) usually implemented in natively in Python, not provided by the OS


python-csp example: Boids
Cohesion 0.030
Avoidance 0.250
Alignment 0.120
Acceleration 0.800

Morals of this story...

- Not every language has nice, high-level concurrency features
- It is still worth porting CSP etc. to your favourite language
  - If you don't like Python, try Actionscript ;-)
- The JVM is not the answer to every ill
- Sometimes waiting is a good idea ...
  - Google will finish Unladen Swallow
  - Jython will get faster (but will it get jythonc back?!)

Future directions

- Mobility
- Performance issues
  - Can we do better?
    - Coroutines, protothreads, ...
    - Unladen Swallow (LLVM -> ???)
- Using the underlying thread / process libraries
  - Brings an overhead
  - Doesn't directly implement POSIX anything
  - May prove useful to replace
- Pythonic issues
  - Get high level concurrency into the standard library ;-)
- Pervasive computing -- bigraphs?