

ACSAC Panel, Monash Uni, February 2002
Where have all the systems students gone?

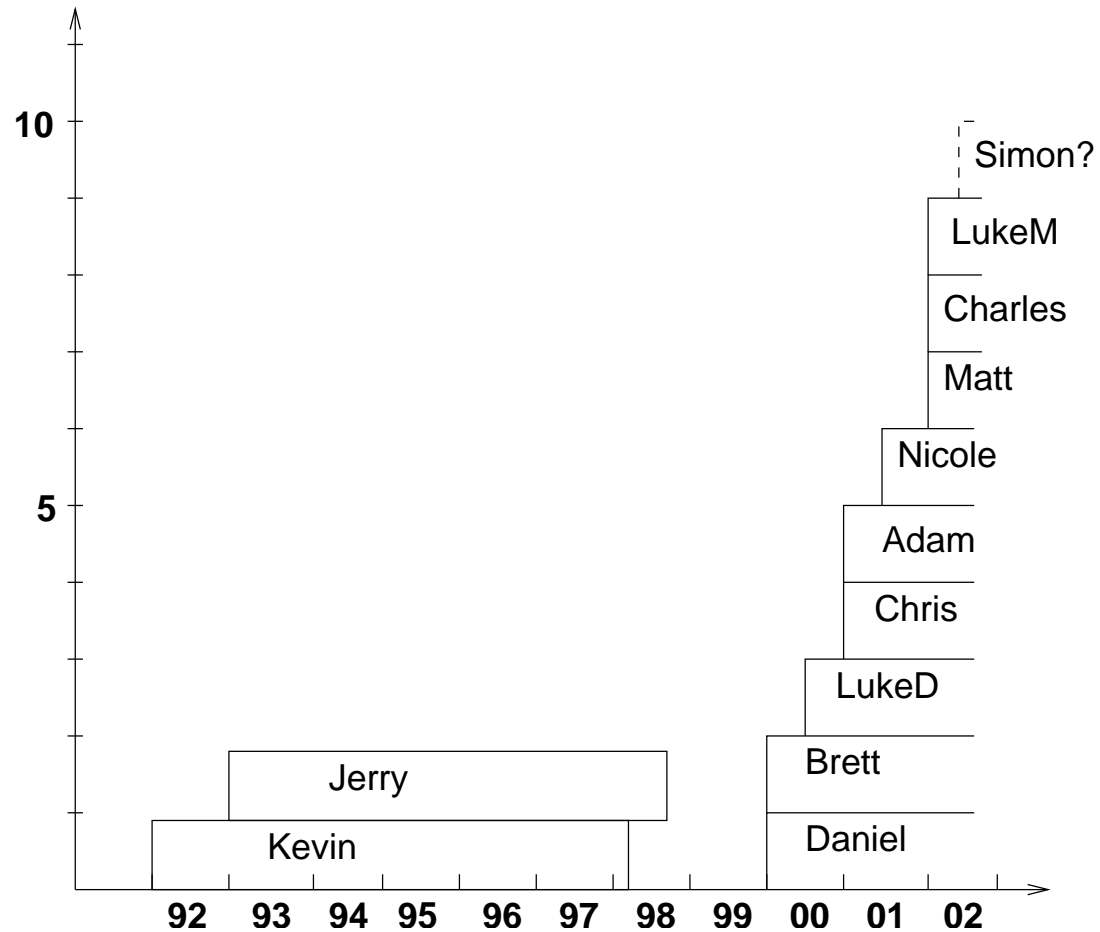
Some provocative answers by
Gernot Heiser
School of Computer Science & Engineering
UNSW

Question: Where have all the students gone?

Assertion 0: They are up for grabs!

...but you must know how to get them.

My Experience: Research Students



What happened?

ASSERTION 1: TO GET STUDENTS INTO RESEARCH YOU NEED:

1. **Research:** to give them a vision
2. **Teaching:** to get them interested and get them trained
3. **Buzz:** get them excited

Research

- Students want to see that something's going on
 - need exciting/visible project that catches students' imagination■
- Chicken & egg problem
 - the ARC sux!
 - looks for long & heavy publication lists (which don't exist in systems)
 - lacks competent reviewers (systems apps get reviewed by people who *think* they know about systems)
 - lacks insight into systems research■
- Critical mass problem
 - need a reasonably big group in order to do anything interesting■

Can only resolve this with a long-term strategy.■

Buzz

- Linux helps

- has put systems back on the agenda
- provides motivation (“Lets show it to Microsoft!”)
- provides training outside classroom■

- The *KEG Lab!*

- put a few machines into a room next to the research students
- get a few talented U/G students in there
- let them play!
 - Outcomes: PLEB boards, Sulima ISA simulator, tools... and interest■

⇒ build up a reputation of “coolness”

- “I’m in KEG hacking kernels, and what do you do?”
- “What do I have to do to get into KEG?”■

Teaching

Assertion 2: Systems teaching is generally broken■

E.g. operating systems:

- OS courses are mostly about concurrency
- Rest is mostly about trivial algorithms and data structures
- What's the message to students?■
 - OS is **boring!**
 - OS is trivial!
 - Why would a sharp student want to do systems research?■

Teaching...

Assertion 3: It is possible to send the *right* message:

- OS is cool!
- OS is challenging!
- I want to be involved!

What To Do — Part 1: Lectures

- OS is **not** about concurrency & trivial scheduling algorithms
 - concurrency is a small part
 - monitors and philosophers don't live in OS kernels
 - the locking problems there are in kernels require too much background
 - disk scheduling is mostly irrelevant (SCSI does it for you)
 - process scheduling is a small topic (worth 1h max)
- OS **is** about:
 - tradeoffs (time vs space, performance vs predictability, fairness vs throughput, portability vs performance, performance vs ease-of-use)
 - performance (which design will work and *why*?)
 - hardware
 - how does interrupt/exception/context switch *really* work?
 - how does a TLB work and what does this mean for page tables?
 - if you aren't showing any assembler code you aren't teaching OS!

What To Do — Part 2: Labs/Assignments

- Questions:
 - What do students learn by playing with a thread package?
 - What do students learn by implementing a FAT inside a UNIX file?■
- What *should* assignments look like?
 - They should be about **OS code!**
 - They *must* force students to write and debug *kernel* code
 - They are *inherently* hard!■
- Don't try to protect students from the harsh reality!
 - You'll get the weak students looking to you for an easy thesis.
 - You'll only see the backs of the high fliers.■

What To Do — Part 3: Hierarchy of Courses

COMP3231 “Operating Systems” (1st session 3rd year, 400 stud)

- students implement page tables and TLB miss handler in microkernel
- “Hardest and best 3rd year course!”■

COMP9242 “Advanced OS” (2nd session 3d year, 20 stud)

- students build complete OS on top of minimal microkernel
 - device driver
 - VM management
 - process management
 - file system
- “Hardest, best and most fun course ever!”■

COMP9243 “Distributed Systems” (4th year, 30 stud)

COMPxxxx “Comparative Computer Architecture” (res. stud)■

Results

- best kernel hackers in the Southern Hemisphere
- excellent understanding of systems issues
- extremely keen to build systems
- regularly send undergraduates or fresh graduates to IBM Watson
- growing group of growing reputation

Conclusions

Systems is in trouble, but...■

- it's unjustified
 - there are plenty of challenges and opportunities
- it's dangerous
 - the country will lose big time if we give up
- we can do it!
 - ... at least at UNSW