# Some Lessons Learned Reviewing Scientific Code

Chris Morris
STFC
SECSE2008

# Nearly silent error handling

```
ServerThread() {
    ....
    try {
        ....
    } catch (IOException ex) {
        ex.printStaceTrace();
    }
    // what postcondition?
}
```

# Hypothesis: Lack of error handling is characteristic of scientific end-user code

- This defect also found in commercial code
- Must define “professional software developer”
- - risk of a circular argument

# Testability

- NPATH complexity of one method:

  770,943,744,005,163,750,045

- Lack of testing is characteristic of the end-user scientific coding process
- System validation may be impractical
- Unit testing is not attempted
- Static analysers not used
- Never seen a job advert for a tester

# Poor use of OO

- “An example of a class with a lot of duplicate code is [...], which has lines copied from (or to) five other classes.”
- Fifteen per cent of [...] is lines that have been copied and pasted.
- [...] has 28 blocks of 100 or more lines that have been copied and pasted.
- 70% of classes have DIT of 0 or 1.
- Also unfamiliar with transactions, postconditions.

# No explicit quality goals

- None of the projects reviewed had a written quality policy
- Appropriate quality goals may not be obvious
- SCHED: robustness
- EXP: recoverability
- LIMS: reliability

# Other findings

- Circular dependencies – no process to preserve architecture
- Numerical stability
- Review process encourages reflection: traceability from process deficiencies to code defects

# Proof of Concept coding

- Goal to show feasibility, not make shrink-wrap product
- Defects matter only if fundamental
- Most SE processes inappropriate
- This is the formative experience of scientific end-user programmers
- But: 2008's prototype may be 2015's clinical application

# Senior Codes

- Long-lived, many KLOC, Fortran, HPC, physics simulations
- Refutable hypotheses:

- the model implemented is the one in the theory document

- the solution method is convergent

- Possible to make unit tests
- SE techniques and tools not fully appropriate