Software Design and Modelling

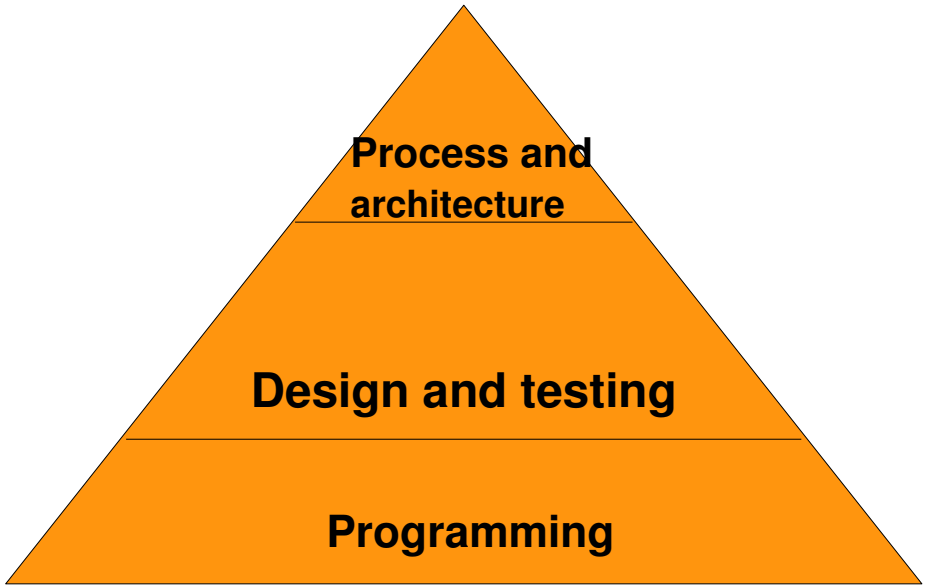
Perdita Stevens

School of Informatics
University of Edinburgh

Plan

- What's this course about?
- How will the course run?
- What are you supposed to know already?

What's this course about?



We assume you can program in Java, given a design.

Aim: after this course, if you understand some requirements you'll be able to develop a good design to satisfy them.

This course goes well with: Software Testing; Software Architecture, Process and Management.

Elephant trap

At university, and in most summer jobs, you see small software systems and work with them over short timeframes.

Elephant trap

At university, and in most summer jobs, you see small software systems and work with them over short timeframes.

In that context, hacking works OK.

Elephant trap

At university, and in most summer jobs, you see small software systems and work with them over short timeframes.

In that context, hacking works OK.

But it does not work at scale!

Elephant trap

At university, and in most summer jobs, you see small software systems and work with them over short timeframes.

In that context, hacking works OK.

But it does not work at scale!

I will try to help you to understand why the techniques we learn in this course are worthwhile, but if you evaluate them against short small experiences, you may not get it.

Try to remember that real-world software systems can be many millions of LOC, many hundreds of person-years of effort, spread over many years, very complex.

Method

Learning to design well is hard.

Method

Learning to design well is hard.

Teaching someone to design well is impossible.

Method

Learning to design well is hard.

Teaching someone to design well is impossible.

But we can teach, e.g.

- the vocabulary of design criteria: what makes a design good?
- how to model designs so that they can be discussed
- how to learn from others' knowledge e.g. recorded as patterns.

How will the course run?

Two lectures and one guided lab most weeks: see schedule page.

Some "flipping": I will often ask you to read/watch videos teaching you basic information outside the timetabled slots, and will then use the timetabled slots to go through examples and let you ask questions.

Piazza forum for questions and discussion.

Assessment

50% lab assessment **in the week 6 lab slot**.

Aims not to be scary, but to check you have kept up and are ready for the second part of the course.

Assessment

50% lab assessment **in the week 6 lab slot**.

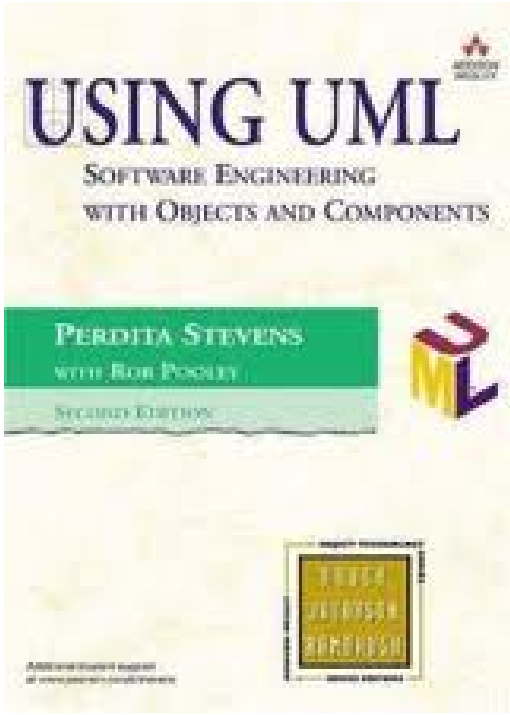
Aims not to be scary, but to check you have kept up and are ready for the second part of the course.

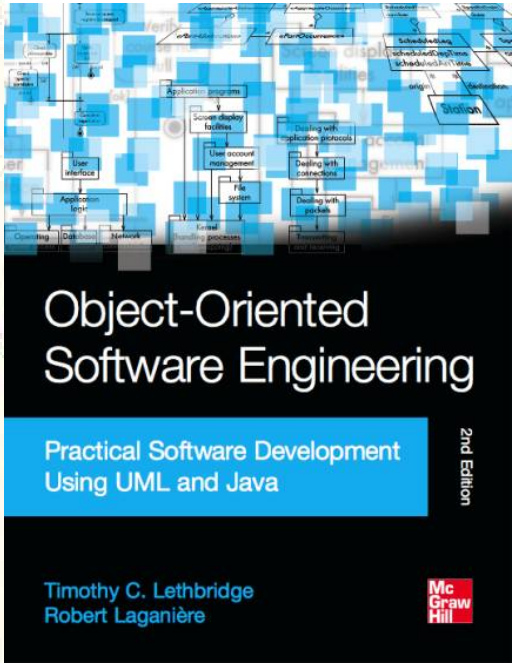
50% written exam in **December**.

See sample paper on course home page.

Format: compulsory question 1, then a choice of 2 questions.

Recommended books





Second-hand copies of UU are fine, but make sure they're second edition (for UML2).

Beyond exam success

80% of success is showing up.

80% of becoming a good software designer is caring and thinking about software design.

From now on, every time you read or write code, ask yourself: why is it designed this way? Could it be improved? How?

What are you supposed to know already?

1. How to program competently in Java (Inf1-OP) – including understanding basic OO concepts.
2. What software engineering involves (Inf2C-SE) – including basic use of UML.

ASAP: please visit the course web page, join the Piazza class, and fill in the preassessment form.