Lecture 1

We discuss *software quality*: satisfying numerous different stakeholders.

We discuss the *software crisis*: too often we fail to satisfy the stakeholders.

We discuss the nature of *software development*: a model-building process.

Stake-holders

- We have many stake-holders:
  - those who commission the software
  - users
    - technical ‘users’ who install it, install new versions and patches, run the backups, etc.
    - end-users who use it in their jobs
  - other programmers
    - who maintain it
    - who port it
    - who re-use it
    - who replace it

Software quality

High quality software must please all stake-holders.

It will be

○ correct	○ extendible
○ robust	○ reusable
○ time- and space-efficient	○ compatible with other software
○ easy to use	○ portable
○ of sufficiently high functionality	○ verifiable
○ secure	○ repairable
○ delivered on schedule	○ delivered on budget

This course looks at how to produce high quality software

Projects often fail

- A huge conceptual gap between business person who has a problem to solve, expressed in terms of people, money, books, sales, customers, etc., and the software engineer who produces a solution expressed in terms of variables, flow of control constructions, etc.
- A poor understanding of software (it’s intangible) and software development (an immature discipline)
- Growing ambitions

Models

> *A model is an artifact that represents a part of a world at some level of abstraction that can be used to explain or predict features of the world that is being modeled.*

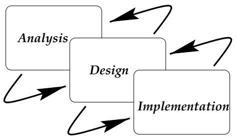
Example:

> A map of some terrain is a model of that terrain.

Programs are models

- Nearly all programs model something.
- E.g.
  - software to assist stock control;
  - software to aid College administration;
  - software to control a factory process;
  - scientific simulation software;
  - games software
- Software development is therefore a model construction process.

Software Development



- Consider an analogy with building construction
- Models get built at each step
- The process is highly iterative

The steps in software development

- *Analysis:* identify a problem, analyse it, specify requirements of the solution (functional requirements & non-functional requirements)
- *Design:* formulate alternative solutions, describe them, rank them & choose between them
- *Implementation:* coding, testing & debugging

Part of the solution to the software crisis

- To help reduce the conceptual gap, our programming language must offer just the right kinds of 'building blocks' for us to build the models we want.
- Some people argue that the majority of programming languages do not adequately do this: these languages are too machine-oriented.
- Languages, such as Java, that fall into the object-oriented paradigm have the right 'building blocks'.