Questions

- Give a definition of Software Engineering?
- Why is Software Engineering necessary?
- Give a number of examples of projects failures.

What is Software Engineering?

- Large, high quality software systems
  - Software engineering techniques are needed because large systems *cannot be completely understood* by one person
  - Identification of missing quality aspects before building
  - Teamwork and co-ordination are required
  - Key challenge: dividing up the work and ensuring that the parts of the system work properly together
  - The end-product must be of high quality

What is Software Engineering?

- Cost, time and other constraints
  - Finite resources
  - The benefit must outweigh the cost
  - Others are competing to do the job cheaper and faster
  - Inaccurate estimates of cost and time have caused many project failures
- Quality attributes:
  - Usability, efficiency, reliability, maintainability, reusability
  - The different qualities can conflict
    - increasing efficiency can reduce maintainability
    - increasing usability can reduce efficiency

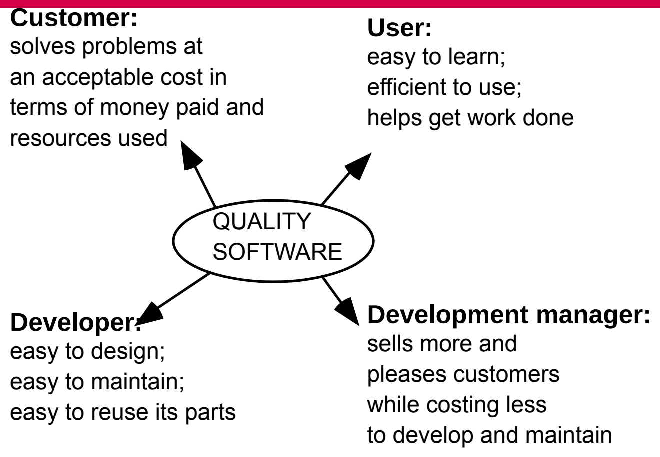
Stakeholders in Software engineering

1. Users
   - Those who use the software
2. Customers
   - Those who pay for the software
3. Software developers
4. Development Managers

Software Quality...

- Usability
  - Users can learn it and fast and get their job done easily
- Efficiency
  - It doesn’t waste resources such as CPU time and memory
- Reliability
  - It does what it is required to do without failing
- Maintainability
  - It can be easily changed
- Reusability
  - Its parts can be used in other projects, so reprogramming is not needed

Software Quality and the Stakeholders



Customer:
solves problems at
an acceptable cost in
terms of money paid and
resources used

User:
easy to learn;
efficient to use;
helps get work done

QUALITY
SOFTWARE

Developer:
easy to design;
easy to maintain;
easy to reuse its parts

Development manager:
sells more and
pleases customers
while costing less
to develop and maintain

Software Quality: Conflicts and Objectives

- The different qualities can conflict
  - Increasing efficiency can reduce maintainability or reusability
  - Increasing usability can reduce efficiency
- Setting objectives for quality is a key engineering activity
  - You then design to meet the objectives
  - Avoids ‘over-engineering’ which wastes money
- Optimizing is also sometimes necessary
  - E.g. obtain the highest possible reliability using a fixed budget

Internal Quality Criteria

- These:
  - Characterize *aspects of the design* of the software
  - Have an effect on the external quality attributes
  - E.g.
    - The amount of commenting of the code
    - The complexity of the code

Short Term Vs. Long Term Quality

- Short term:
  - Does the software *meet the customer's immediate needs*?
  - Is it sufficiently efficient for the volume of data we have *today*?
- Long term:
  - Maintainability
  - Customer's future needs
  - Scalability: Can the software handle larger volumes of data?

Activities Common to Software Projects

- Requirements and specification
  - Domain analysis
  - Defining the problem
  - Requirements gathering
    - Obtaining input from as many sources as possible
  - Requirements analysis
    - Organizing the information
  - Requirements specification
    - Writing detailed instructions about how the software should behave

Activities Common to Software Projects

- Design
  - Deciding how the requirements should be implemented, using the available technology
  - Includes:
    - *Systems engineering*: Deciding what should be in hardware and what in software
    - *Software architecture*: Dividing the system into subsystems and deciding how the subsystems will interact
    - *Detailed design* of the internals of a subsystem
    - *User interface design*
    - *Design of databases*

Activities Common to Software Projects

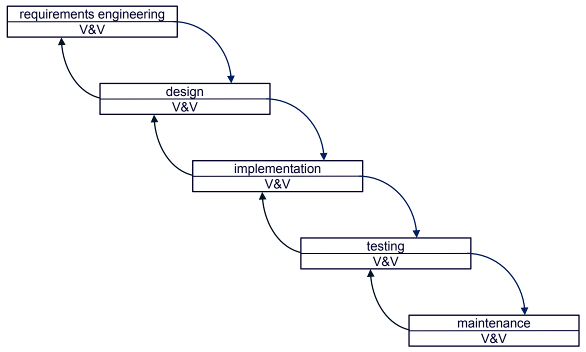
- Modeling
  - Creating representations of the domain or the software
    - Use case modeling
    - Structural modeling
    - Dynamic and behavioral modeling
- Programming
- Quality assurance
  - Reviews and inspections
  - Testing
- Deployment & maintenance
- Managing the process

Software Engineering Projects

- Most projects are *evolutionary* or *maintenance* projects, involving work on *legacy* systems
  - Corrective projects: fixing defects
  - Adaptive projects: changing the system in response to changes in
    - Operating system
    - Database
    - Rules and regulations
  - Enhancement projects: adding new features for users
  - Reengineering or perfective projects: changing the system internally so it is more maintainable

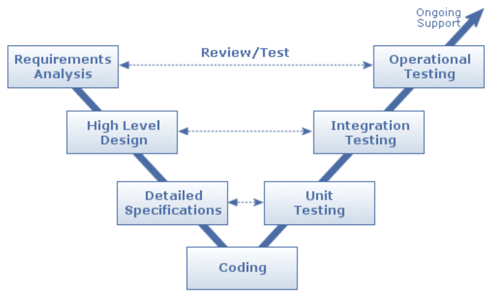
Software development model

Waterfall model



Software development model

V-model


Ongoing Support
Requirements Analysis
Review/Test
Operational Testing
High Level Design
Integration Testing
Detailed Specifications
Unit Testing
Coding