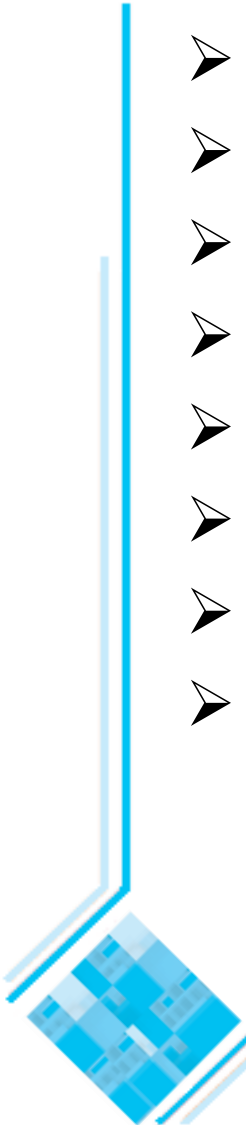


# Object-Oriented Software Engineering Course

- **Software and Software Engineering**
- **Review to Object Oriented**
- **Basic Software Development on Reusable Component**
- **Development Requirements**
- **Modelling with class**
- **Modelling interaction and Behaviors**
- **Architecture and Design Software**
- **Manage the Software Process**



# Object-Oriented Software Engineering

3Th Software Engineering

## **Chapter 1: Software and Software Engineering**



# 1.1 The Nature of Software...

## **Software is intangible**

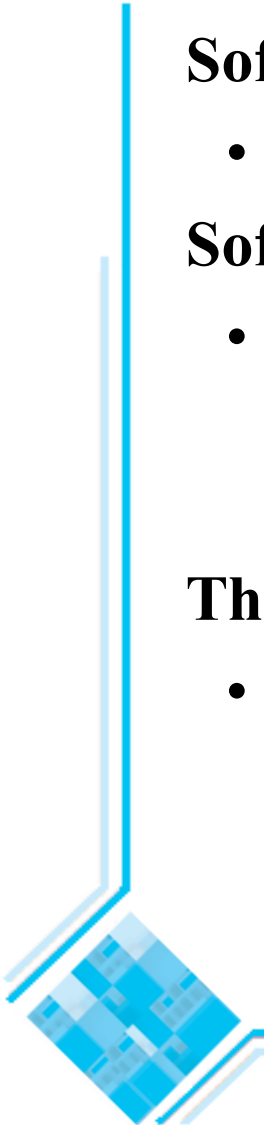
- Hard to understand development effort

## **Software is easy to reproduce**

- Cost is in its *development*
  - in other engineering products, manufacturing is the costly stage

## **The industry is labor-intensive**

- Hard to automate



# The Nature of Software ...

## **Untrained people can hack something together**

- Quality problems are hard to notice

## **Software is easy to modify**

- People make changes without fully understanding it

## **Software does not ‘wear out’**

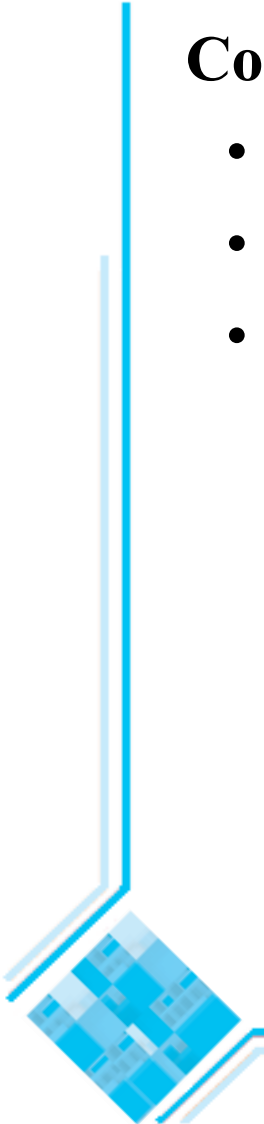
- It *deteriorates* by having its design changed:
  - erroneously, or
  - in ways that were not anticipated, thus making it complex



# The Nature of Software

## Conclusions

- Much software has poor design and is getting worse
- Demand for software is high and rising
- We have to learn to ‘engineer’ software



# Types of Software...

## **Custom**

- For a specific customer

## **Generic**

- Sold on open market
- Often called
  - COTS (Commercial Off The Shelf)
  - Shrink-wrapped

## **Embedded**

- Built into hardware
- Hard to change



# Types of Software

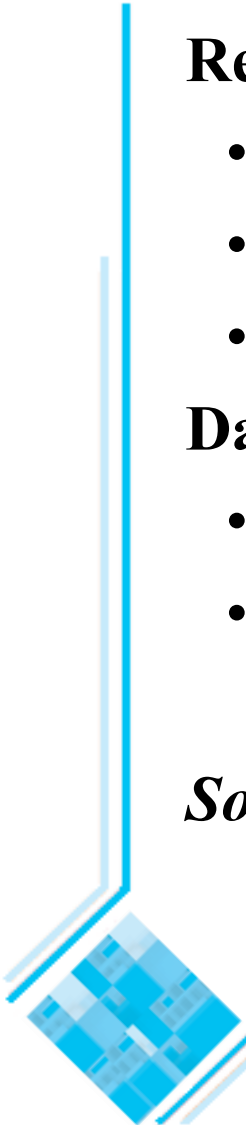
## **Real time software**

- E.g. control and monitoring systems
- Must react immediately
- Safety often a concern

## **Data processing software**

- Used to run businesses
- Accuracy and security of data are key

*Some software has both aspects*



# 1.2 What is Software Engineering?...

**The process of solving customers' problems by the systematic development and evolution of large, high-quality software systems within cost, time and other constraints**

## **Other definitions:**

- IEEE: (1) the application of a systematic, disciplined, quantifiable approach to the development, operation, maintenance of software; that is, the application of engineering to software. (2) The study of approaches as in (1).
- The Canadian Standards Association: The systematic activities involved in the design, implementation and testing of software to optimize its production and support.

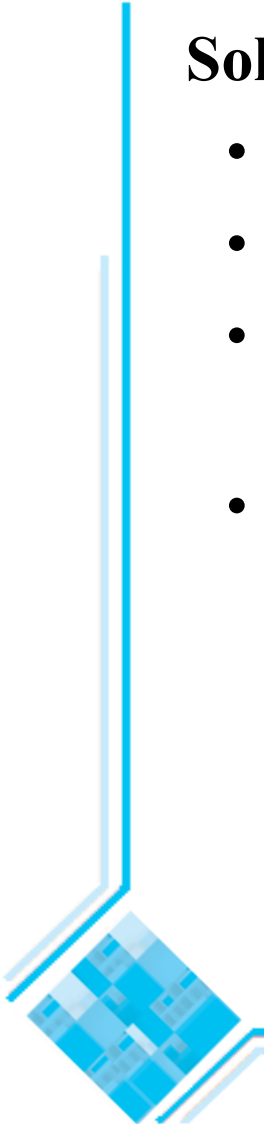




# What is Software Engineering?...

## Solving customers' problems

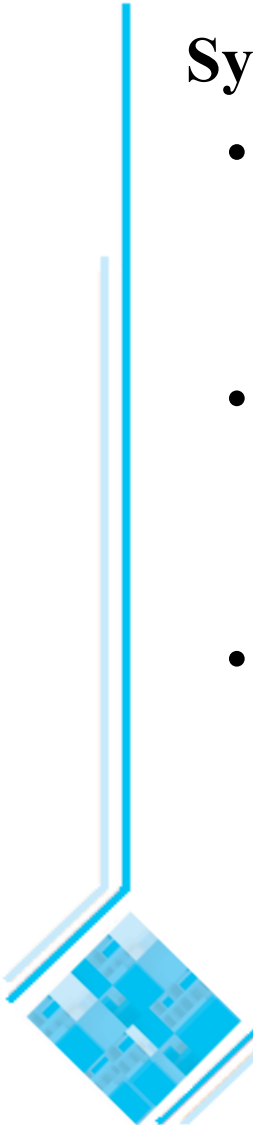
- This is the *goal* of software engineering
- Sometimes the solution is to *buy, not build*
- Adding unnecessary features does not help solve the problem
- Software engineers must *communicate effectively* to identify and understand the problem



# What is Software Engineering?...

## Systematic development and evolution

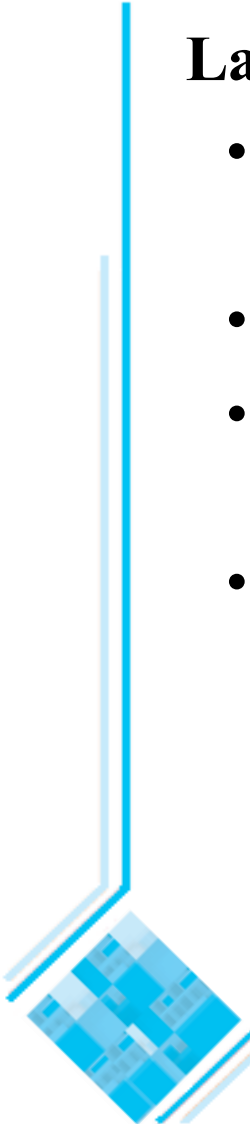
- An engineering process involves applying *well understood techniques* in a organized and *disciplined* way
- Many well-accepted practices have been formally standardized
  - e.g. by the IEEE or ISO
- Most development work is *evolution*



# What is Software Engineering?...

## **Large, high quality software systems**

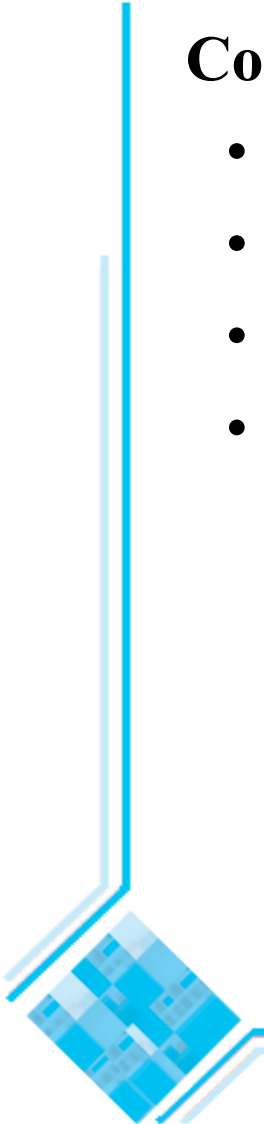
- Software engineering techniques are needed because large systems *cannot be completely understood* by one person
- Teamwork and co-ordination are required
- Key challenge: Dividing up the work and ensuring that the parts of the system work properly together
- The end-product must be of sufficient quality



# 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



# Stakeholders in Software Engineering

## **1. Users**

- Those who use the software

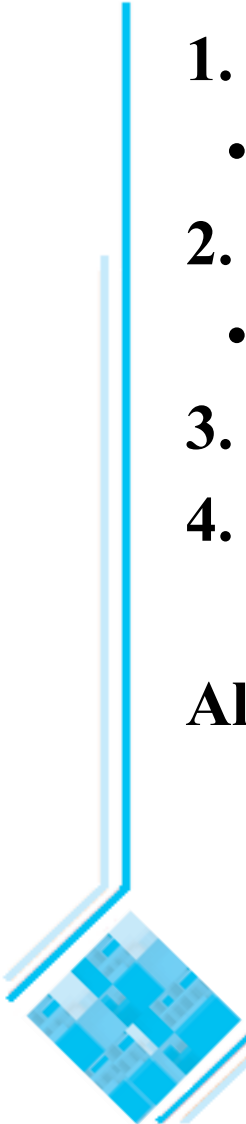
## **2. Customers**

- Those who pay for the software

## **3. Software developers**

## **4. Development Managers**

**All four roles can be fulfilled by the same person**



# 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 Engineering Projects

**Projects that involve building on a *framework* or a set of existing components.**

- A framework is an application that is missing some important details.
  - E.g. Specific rules of this organization.
- Such projects:
  - Involve plugging together *components* that are:
    - Already developed.
    - Provide significant functionality.
  - Benefit from reusing reliable software.
  - Provide much of the same freedom to innovate found in green field development.



# Activities Common to Software Projects...

## Requirements and specification

- Includes
  - 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 behavioural modeling

## **Programming**

## **Quality assurance**

- Reviews and inspections
- Testing

## **Deployment**

## **Managing the process**

