CS2704: Object Oriented Software Design

Topic 1: Introduction
Dr. Ben Keller
Fall 1999

Outline
• Procedural versus object-oriented design
• Object-oriented design strategies
• Course overview

Procedural Programming
• Problem is divided into sequence of sub-problems to be solved
• Program is sequence of procedure calls
• Think in terms of tasks and subtasks
• Languages: C, Pascal, Fortran, COBOL, etc

Procedural Design
• Key: identify simple tasks that can be programmed easily
• Top-down: break tasks into subtasks
• Bottom-up: write small useful routines and compose into larger procedures
• Diagrammed as structure charts

Problems in Procedural Programs
• Large program made up of many small procedures
• Not clear which does what to what data
• No enforced control over access to data
• Difficult to fix bugs, modify, and use procedures in other programs

Software Engineering Goals
• Reusability – components can be used in many applications
• Extensibility – ease of change
• Flexibility – modifications do not “break” system
Object-Oriented Programming
- Think of building program from parts
  Like building a machine!
- Parts are objects that interact to solve problem
- Define classes of objects that can be reused
- Think in terms of objects and interactions
- Languages: C++, Java, Eiffel, Smalltalk, etc

Why Object-Oriented?
- Object interactions are defined by methods, which are just procedures
- Programs still sequences of “procedure” calls, but think of as interacting objects
- Procedures organized around objects
  - Data access easier to understand
  - Data access easier to control
- “Easier” to avoid design problems

Generalization in OOP
- Two approaches to defining classes in terms of others
  - Inheritance – “inherit” properties of other classes
  - Parameterized classes (templates) – class defined in terms of parameter classes
- Design patterns are solutions to common design problems

Object-Oriented Design
- Identify objects and classes strategies: abstraction and separation
- Identify how objects interact in system strategy: composition
- Identify hierarchies of related classes strategy: generalization

Object-Oriented Design Strategies
- Abstraction – modeling essential properties
- Separation – treat what and how independently
- Composition – building complex structures from simpler ones
- Generalization – identifying common elements

Connections