Object Model

Object Orientated Analysis and Design

Benjamin Kenwright

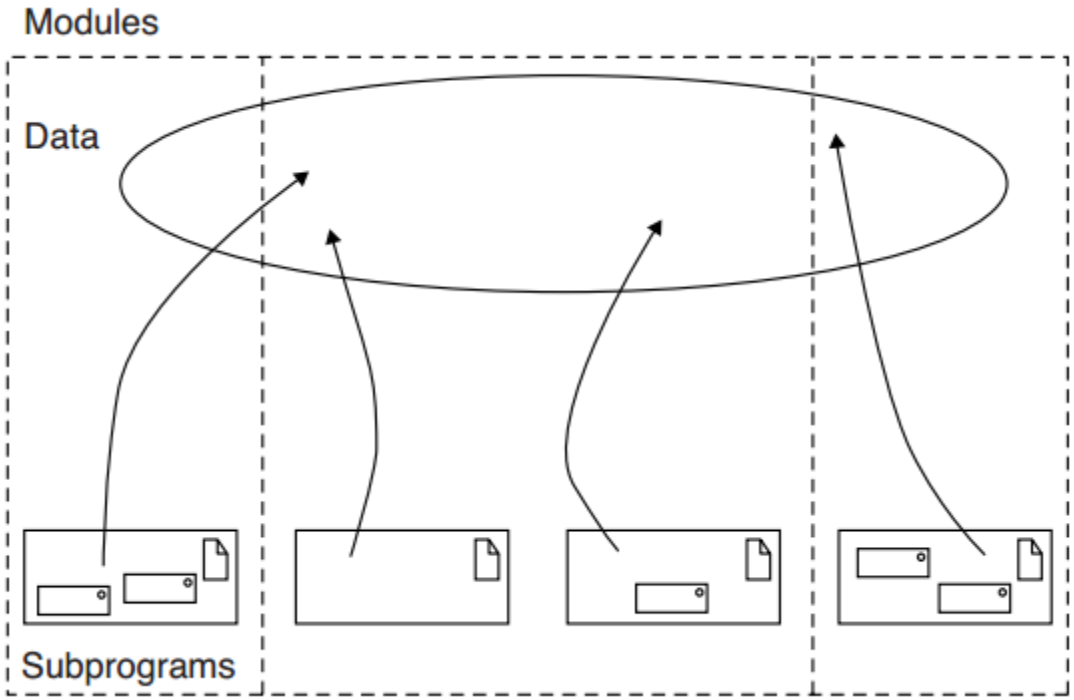
Outline

- What do we mean by the Object Model?

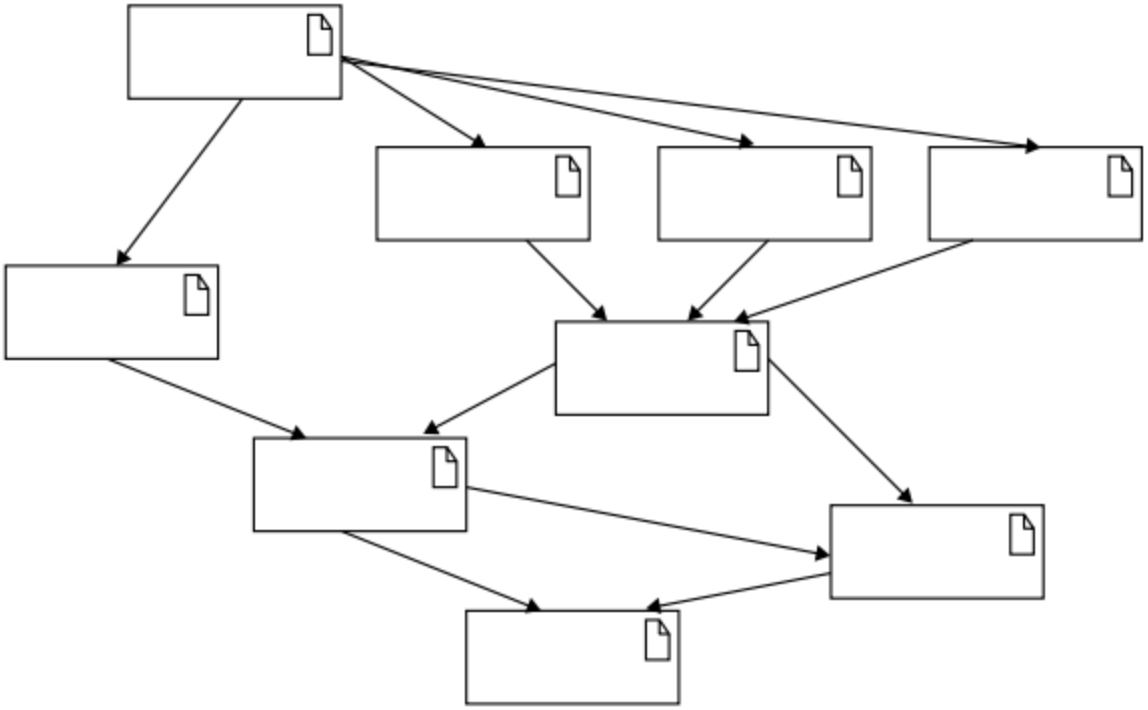
Evolution of the Object Model

- Where did the Object Model come from?

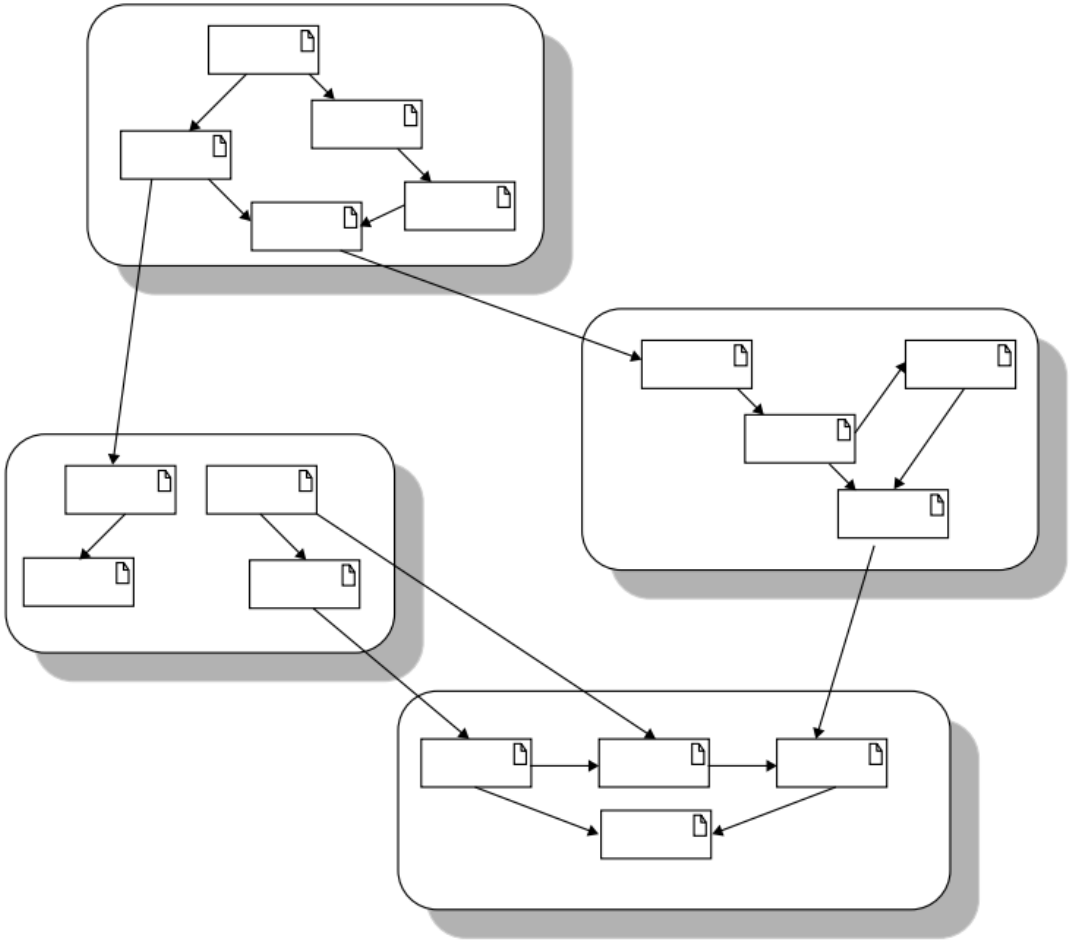
Topology of Early Generation Programming Languages



Topology of Small to Moderate-Sized Applications Using Object-Based and Object-Oriented Programming



Topology of Large Applications Using Object-Based and Object-Oriented Programming Languages



Foundation of the Object Model

- Object-oriented design methods have evolved to help developers *exploit* the expressive power of *object-based* and object-oriented programming *languages*
- *Classes and Objects as basic building blocks*

What is Object-Oriented Programming (OOP)?

What is Object-Oriented Programming (OOP)?

- Three important parts :
- (1) Object-oriented programming uses objects, not algorithms, as its fundamental logical building blocks (the “part of” hierarchy;
- (2) each object is an instance of some class; and
- (3) classes may be related to one another via inheritance relationships (the “is a” hierarchy).

All Three Elements

- A program may appear to be object-oriented, but if any of these three elements is missing, it is *not* an object-oriented program
- For example:
  - Programming *without inheritance* is distinctly not object oriented; that would merely be programming with abstract data types

Question

- What are the three important parts of Object-Oriented Programming (OOP)?

A. uses objects; each object is an instance of some class; classes may be related to one another via inheritance
B. use modules; hierarchical structure; structures must be related to one another via inheritance
C. hierarchical structure; collection of objects; objects must be related to one another via polymorphism

Answer

- A. uses objects; each object is an instance of some class; classes may be related to one another via inheritance

OO Requirements

- Language is object-oriented if and *only* if it satisfies the following requirements:

1. It supports objects that are data abstractions with an interface of named operations and a hidden local state
2. Objects have an associated type [class]
3. Types [classes] may inherit attributes from supertypes [superclasses]

What is Object Oriented Design (OOD)?

What is Object Oriented Design (OOD)?

- Two important parts of object-oriented design:
- (1) leads to an object-oriented decomposition and
- (2) uses different notations to express different models of the logical (class and object structure) and physical (module and process architecture) design of a system

What is Object-Oriented Analysis (OOA)?

- Object-oriented analysis is a method of analysis that examines requirements from the perspective of the classes and objects found in the vocabulary of the problem domain

How are OOA, OOD, and OOP related?

How are OOA, OOD, and OOP related?

- The products of object-oriented analysis serve as the models from which we may start an object-oriented design
- The products of object-oriented design can then be used as blueprints for completely implementing a system using object-oriented programming methods

Object-Orientated Model

- Four *major* elements of this model are:

1. Abstraction
2. Encapsulation
3. Modularity
4. Hierarchy

By *major*, we mean that a model *without any one* of these elements is *not object oriented*

Object-Orientated Model

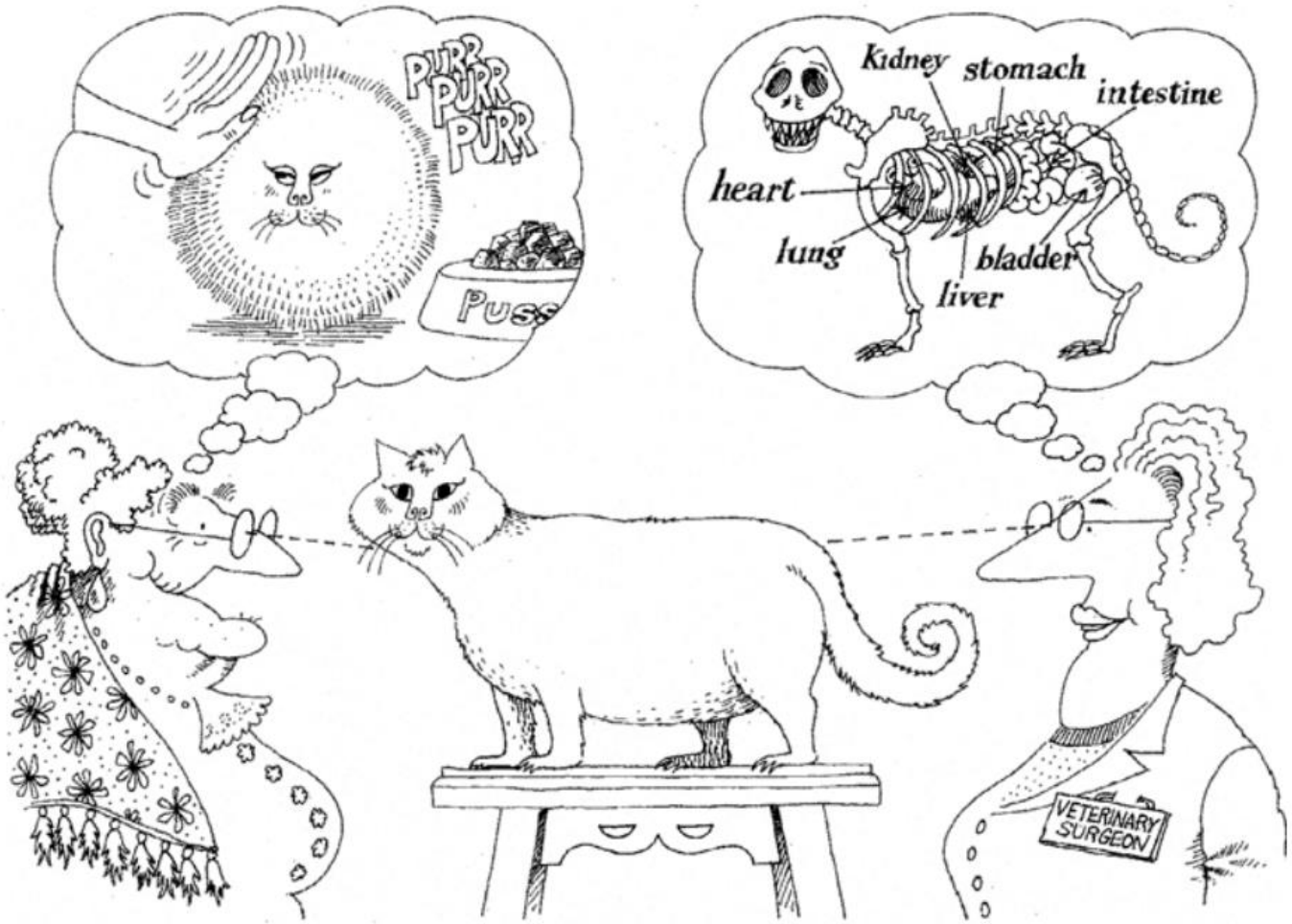
- The three *minor* elements of the object model:

1. Typing
2. Concurrency
3. Persistence

By minor, we mean that each of these elements is a *useful*, but *not essential*, part of the object model

What is the Meaning of Abstraction?

What is the Meaning of Abstraction?



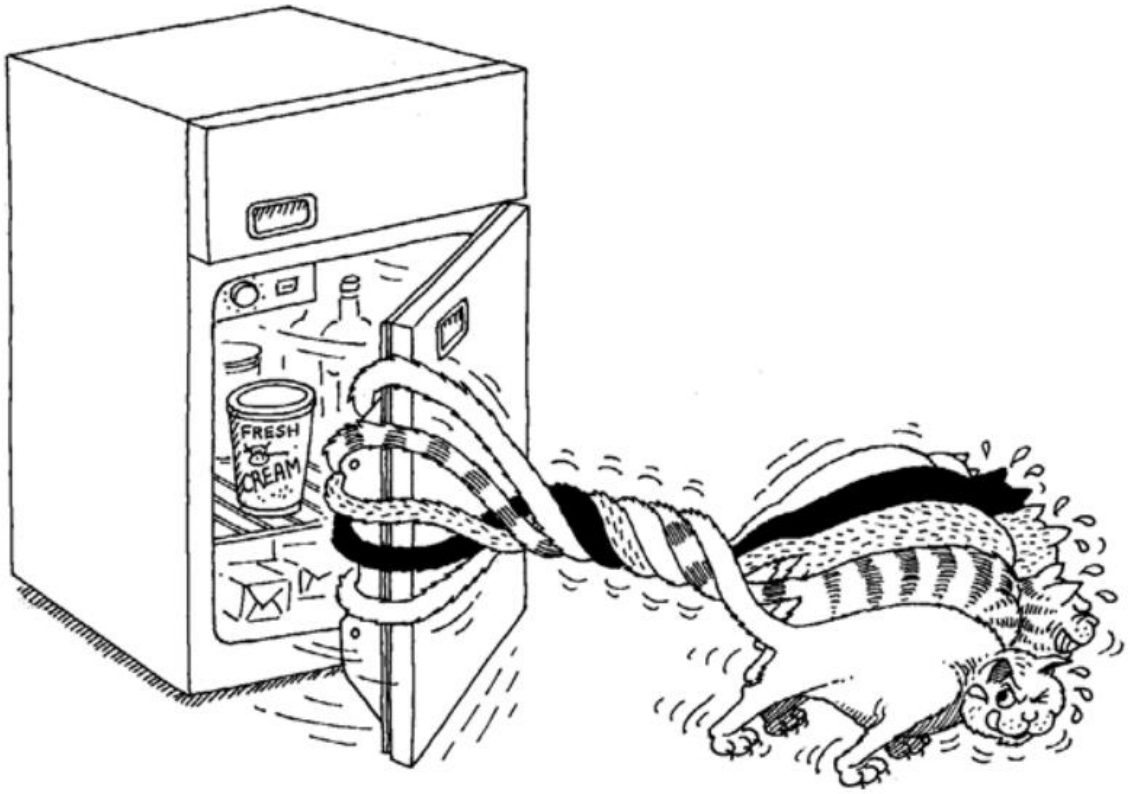
Abstraction focuses on the essential characteristics of some object, relative to the perspective of the viewer.

What is the Meaning of Abstraction?

- An abstraction denotes the essential characteristics of an object that distinguish it from all other kinds of objects and thus provide crisply defined conceptual boundaries, relative to the perspective of the viewer
- Abstraction of an object should precede the decisions about its implementation

Synergy

- No object stands alone;
- Every object collaborates with other objects to achieve some behavior



Objects collaborate with other objects to achieve some behavior.

What is the Meaning of Encapsulation?

What is the Meaning of Encapsulation?

- Encapsulation is achieved through information hiding (not just data hiding), which is the process of hiding all the secrets of an object that do not contribute to its essential characteristics

Abstraction & Encapsulation

- Abstraction and encapsulation are complementary concepts
- *Abstraction focuses* on the observable behavior of an object,
- whereas *encapsulation focuses* on the implementation that gives rise to this behaviour
- Encapsulation provides explicit barriers among different abstractions and thus leads to a clear separation of concerns

Abstraction & Encapsulation Cont.

- For abstraction to work, implementations must be encapsulated
- Each class must have two parts:
  - an interface and an implementation.
- The interface of a class captures only its outside view, encompassing our abstraction of the behavior common to all instances of the class

What is the Meaning of Modularity?

What is the Meaning of Modularity?

Modularity packages abstractions into discrete units.

What is the Meaning of Modularity?

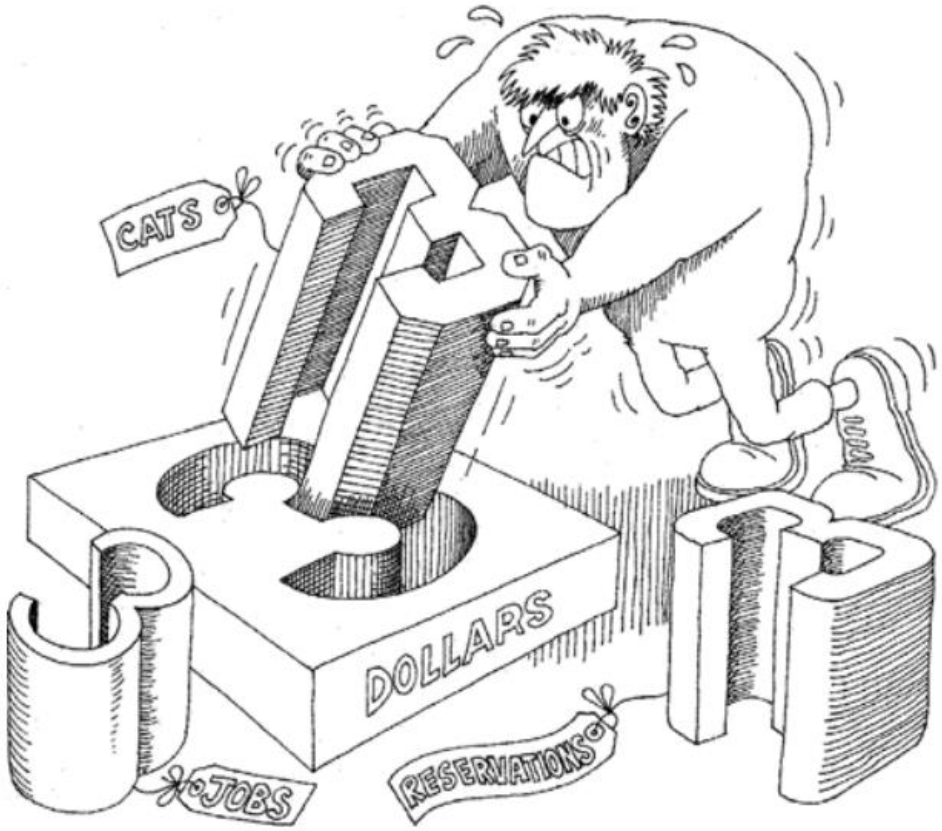
- Act of partitioning a program into individual components can reduce its complexity
- Creates well-defined, documented boundaries (or interfaces) within the program
- Modules are essential to help manage complexity

What is the Meaning of Hierarchy?

What is the Meaning of Hierarchy?

- Hierarchy is a ranking or ordering of abstractions

What is the Meaning of Typing?



Strong typing prevents mixing of abstractions.

What is the Meaning of Typing?

- Typing is the enforcement of the class of an object, such that objects of different types may not be interchanged, or at the most, they may be interchanged only in very restricted ways
- Typing lets us express our abstractions
- Strong typing prevents mixing of abstractions

What is the Meaning of Typing?

- For example, an object may have both a class and a type. In Smalltalk, objects of the classes SmallInteger, LargeNegativeInteger, and LargePositiveInteger are all of the same type, Integer

Typing

- Some important *benefits* to be derived from using strongly typed languages:
  - Without type checking, a program in most languages can ‘crash’ in mysterious ways at runtime
  - In most systems, the edit-compile-debug cycle is so tedious that early error detection is indispensable
  - Type declarations help to document programs
  - Most compilers can generate more efficient object code if types are declared

Static and Dynamic Typing

- Static typing (also known as static binding or early binding) means that the types of all variables and expressions are fixed at the time of compilation;
- Dynamic typing (also known as late binding) means that the types of all variables and expressions are not known until runtime

Benefits of the Object Model

- Many people who have no idea how a computer works find the idea of object-oriented systems quite natural
- Object model reduces the risks inherent in developing complex systems
- Object model produces systems that are built on stable intermediate forms, which are more resilient to change
- Object model encourages the reuse not only of software but of entire designs
- Object model helps us to exploit the expressive power of object-based and object-oriented programming languages

Summary

- Clear idea of Object Model in Object Orientated Analysis and Design
- Address the issues of programming-in-the-large
- Abstraction, Encapsulation, Modularity and Hierarchy
- Typing is the enforcement of the class of an object interchanging (strict rules)

This Week

- Review Slides
- Read Chapter 3

Questions/Discussion