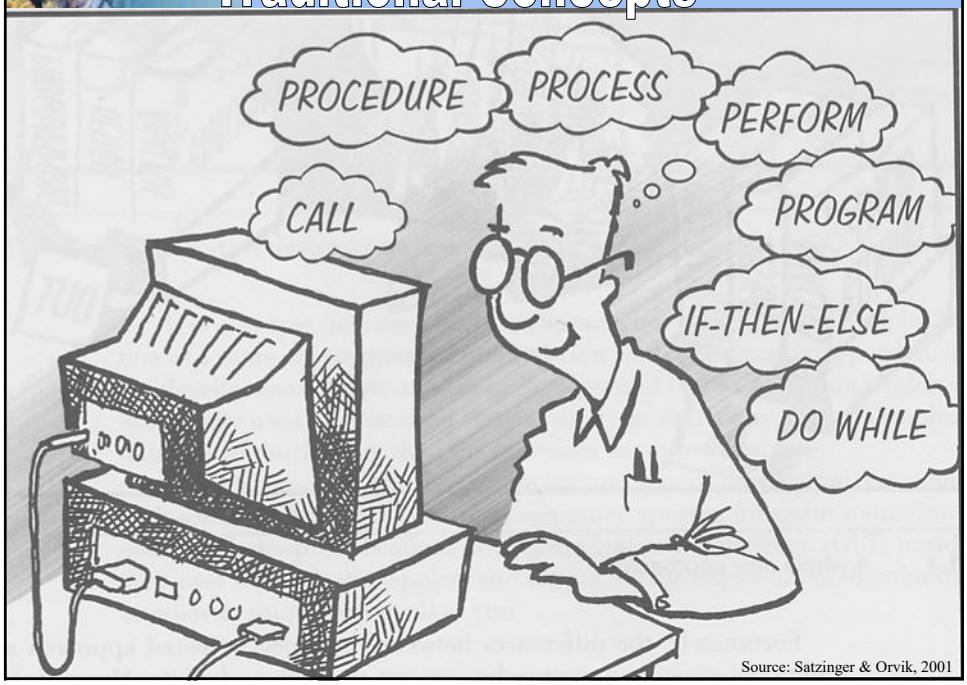
Introduction to OO concepts

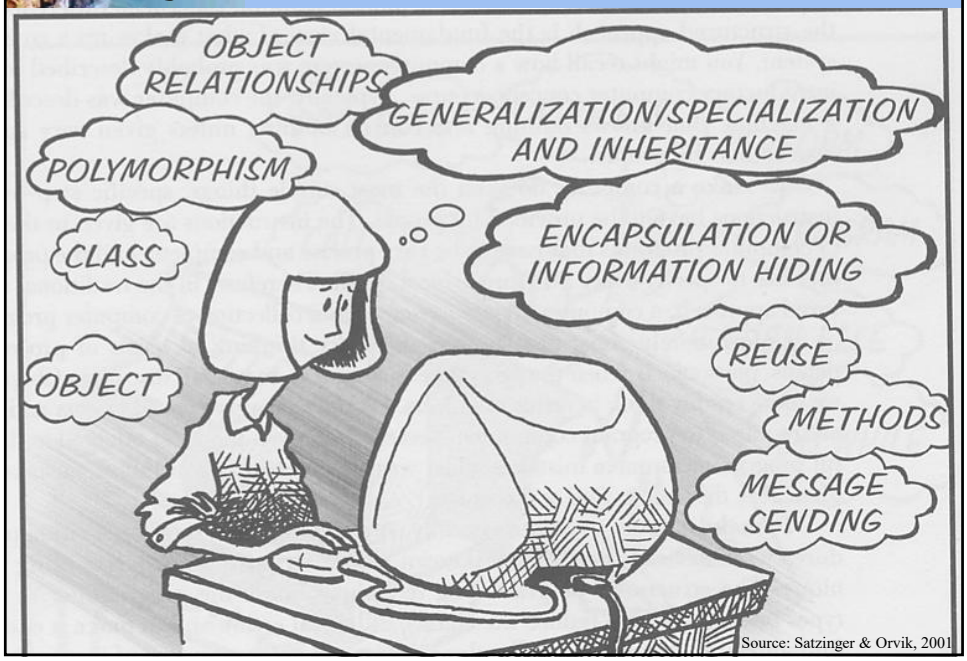
Y

Traditional Concepts



Source: Satzinger & Orvik, 2001

Object concepts to think about!!



Source: Satzinger & Orvik, 2001

Traditional vs. Object-Oriented Approach

- Traditional view
  - an algorithm perspective
  - main building block is the procedure or function
- OO view
  - main building block is the object or class
    - an object is a thing
    - a class is a description of a set of common objects

Remember……..

- A computer system is viewed as a collection of interacting objects.
- Objects are viewed as things that have features (attributes) and exhibit behavior.
- Objects can be grouped and classified
- Objects interact, therefore affect one another – they collaborate…

SO WHAT'S AN OBJECT?

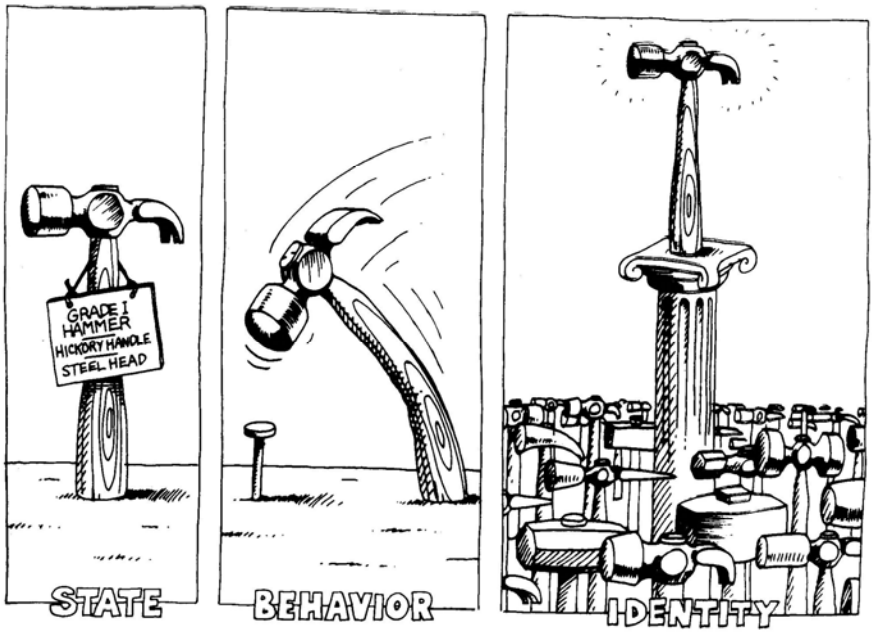
- Anything, real or abstract, about which we store data
- Examples
  - an invoice, an organization, a screen with which a user interacts, a drawing, an airplane, an order-filling process...

WHAT'S AN OPERATION?

- An activity that reads or manipulates data of an object
- Examples
  - calculating a total, checking a balance, adding a new employee, changing an address, deleting a customer ...

Object

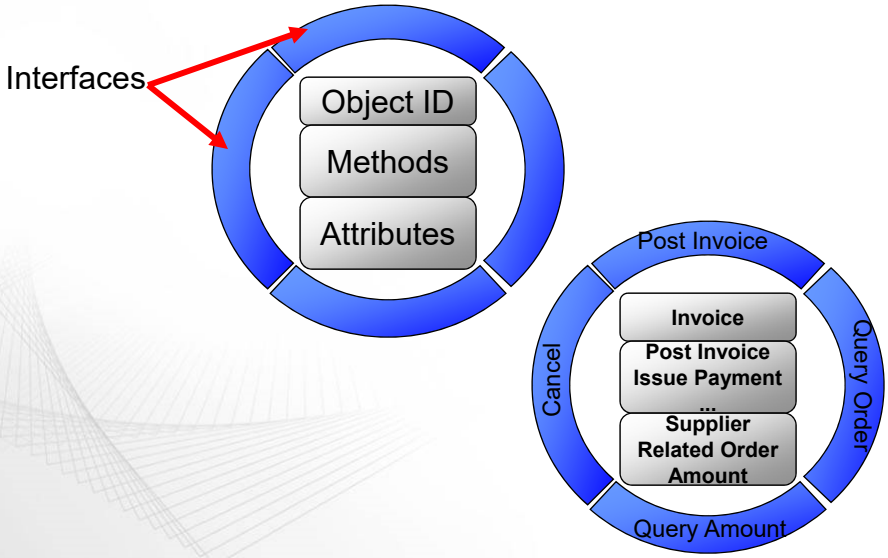
- In an object-oriented system, everything is an object
- Every object has
  - Identity
    - you can name it or distinguish it from other objects
  - State
    - there's generally some data associated with it
  - Behavior (Operate)
    - you can do things to the object
    - the object can do things to other objects (collaborate)



An object has state, exhibits some well-defined behavior, and has a unique identity.

Source: Booch, 1994, 2007

What is an Object?



OBJECT INSTANCES

- An object instance is an example of an object type
- Examples
  - John P. Smith, Invoice #12356
- And just to make it all really confusing, object instances are sometimes called objects!

Used with permission from Pearson Technologies

Class

Objects can be grouped and classified

- Objects are grouped in classes
- Classes are used to distinguish one type of object from another
- A class is a set of objects that share some common structures and behaviors
  - Each object is an *instance* of a class
- In an OO system, behavior of an object is defined by its class

Adapted from: Keng Siau-University of Nebraska-Lincoln

Attribute

Objects are viewed as things that have features (attributes)

- Attributes are *properties* of an object
  - E.g., color, cost, make, and model of a car object
- Properties represent the state of an object

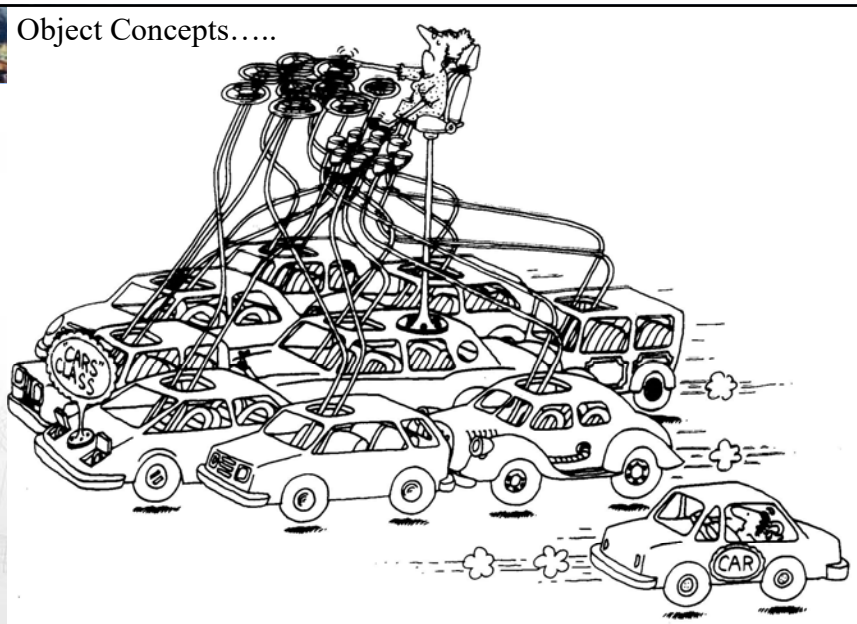
Adapted from: Keng Siau-University of Nebraska-Lincoln

Objects

review

- Class
  - Implementation of an object or set of objects with a common structure and behavior
- Class diagram
  - A tool displaying a hierarchy of classes, including superclasses and subclasses
    - Lab 4 – more coming
- Instance
  - A specific occurrence of an object
- Attributes
  - The properties of an individual object

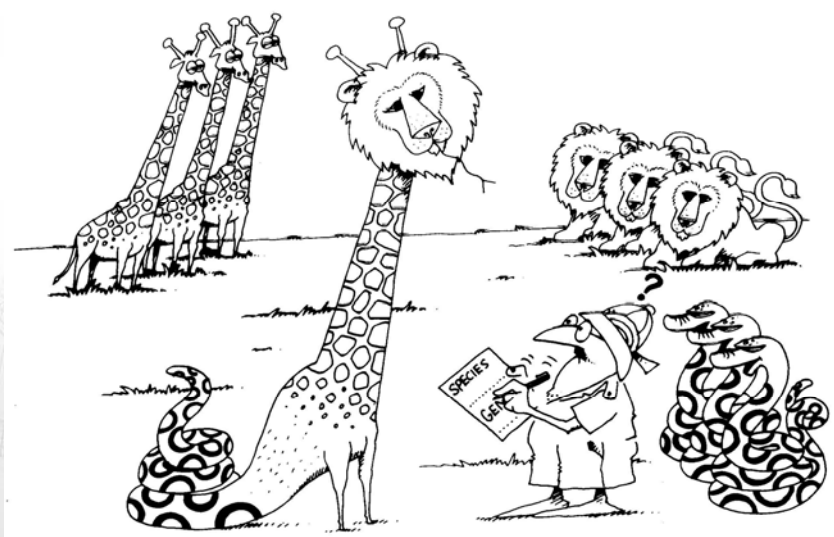
Object Concepts…..



A class represents a set of objects that share a common structure and a common behavior.

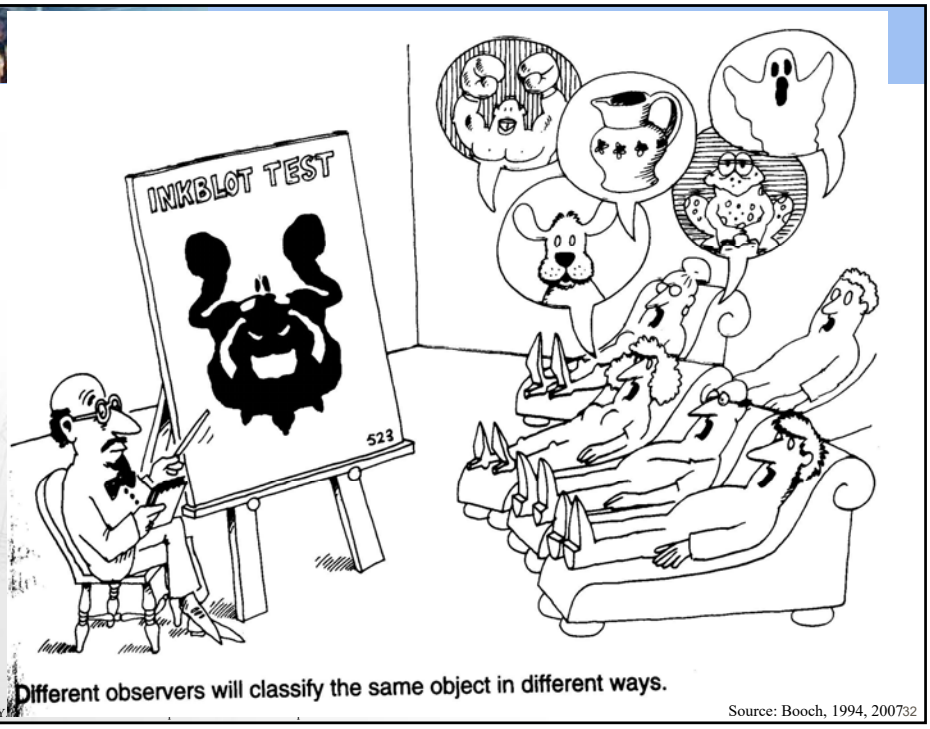
Source: Booch, 1994, 2007

Classification – a set of objects with at least one Common feature.



Classification is the means whereby we order knowledge.

Source: Booch, 1994, 2007



Object Behavior

***Objects** are viewed as things that exhibit **behavior**.*

- **Object behavior is described in procedures (operations) or *methods****
- **A method is defined for a class and can access the internal state of an object of that class to perform some operation**

Adapted from: Keng Siau-University of Nebraska-Lincoln

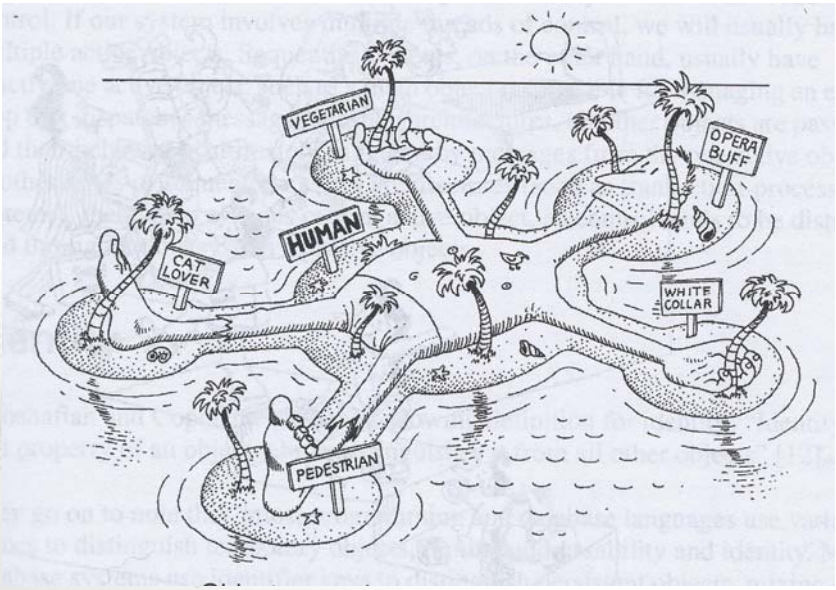
Object Behavior

- Behavior denotes the collection of methods that abstractly describes what an object is capable of doing
  - Each method defines and describes a particular behavior of an object
- Objects take responsibility of their own behavior
  - E.g., an employee object knows how to compute its salary

Adapted from: Keng Siau-University of Nebraska-Lincoln

34

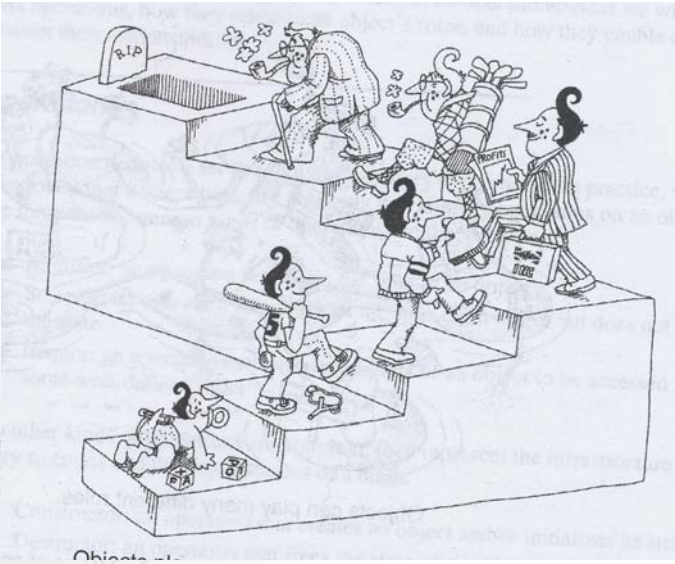
Objects can play many different roles



Source: Booch, 1994, 2007

35

Objects play many different roles During their lifetimes



Source: Booch, 1994, 2007

Message

- Objects perform operations in response to messages
- A message triggers behavior!
- Message is different from a subroutine call
  - Different objects can respond to the same message in different ways
    - e.g., cars, motorcycles, and bicycles will all respond to a stop message -- but differently
- A message has a name

Adapted from: Keng Siau-University of Nebraska-Lincoln

Message

- An object understands a message when it can match the message to a method that has the same name as the message
- Message differs from function
  - Function says how to do something
  - Message says what to do
- Message Exchange:
  - Can trigger procedures or operations
  - Can cause an object to change state

 38

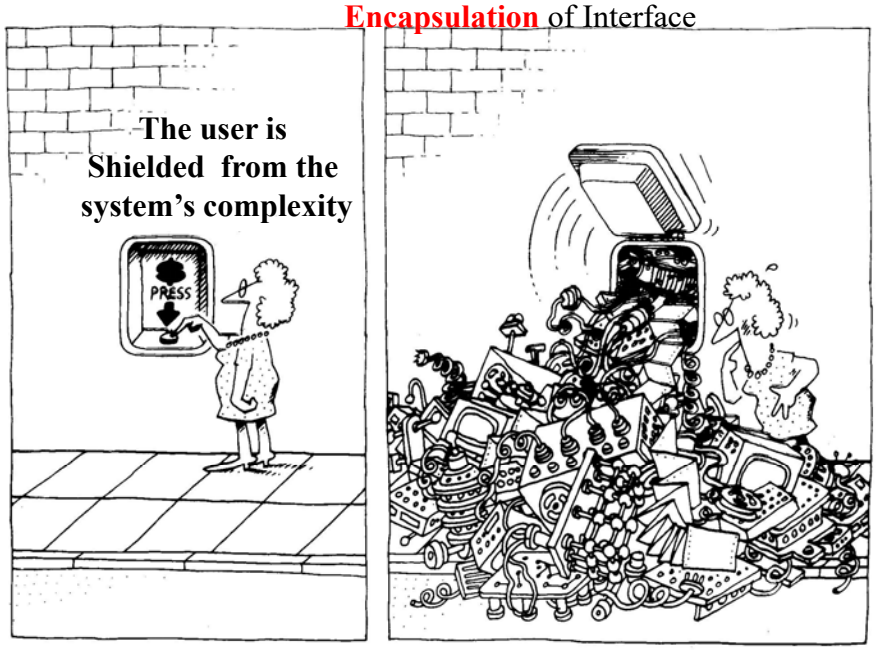
Difference Between Methods and Messages

- Message is the instruction
- Method is the implementation

- E.g., making French onion soup
  - Telling someone to make the soup is the message
  - The way the French onion soup is prepared is the method
  - The French onion soup is the object

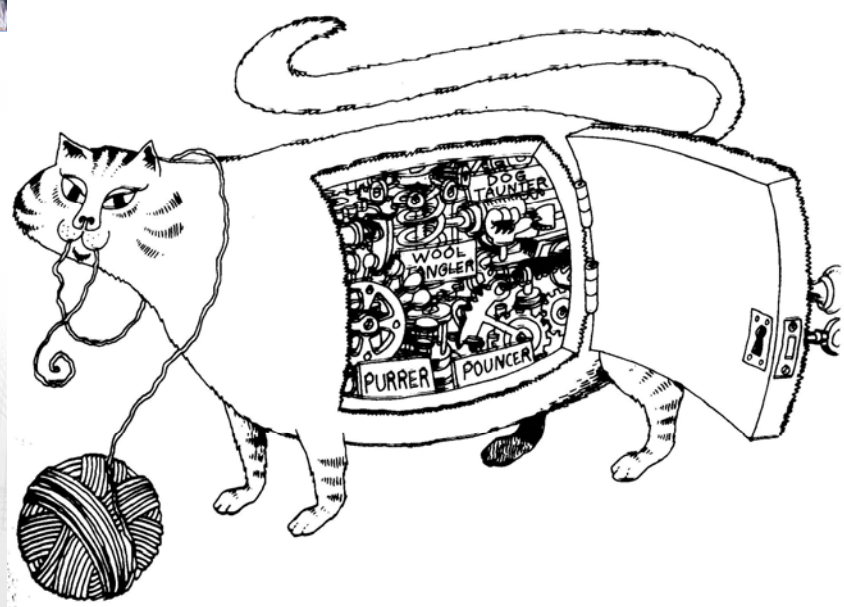
Adapted from: Keng Siau-University of Nebraska-Lincoln 39

Encapsulation of Interface



The task of the software development team is to engineer the illusion of simplicity.

Source: Booch, 1994, 2007



Encapsulation hides the details of the implementation of an object.

Source: Booch, 1994

Encapsulation

- Example: A single software component to lookup student account # in a student database
- The purposeful hiding of information, thereby reducing the amount of details that need to be remembered/communicated among programmers.
- Encapsulation is a form of information hiding
  - but they are slightly different – we will get into this more later in the semester.
    - Encapsulation: a language facility
    - Information Hiding: a design principle
- An object encapsulates the data and methods
  - Users cannot see the inside of the object "capsule," but can use the object by calling the object's methods
  - No object can operate directly on another object's data

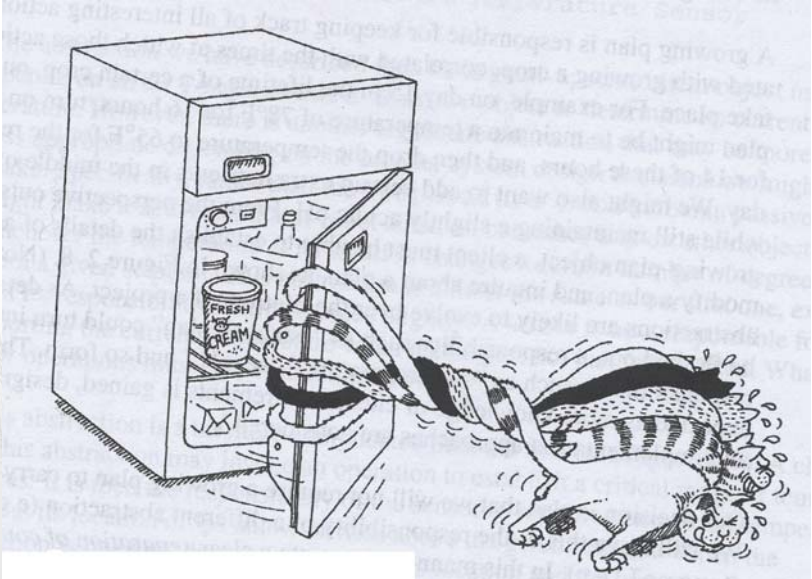
 42

Polymorphism

- A term closely related to message sending is Polymorphism, which literally means multiple forms.
- The same operation may behave differently on different classes
- The ability to send the same message to different object classes.
  - and have the message trigger the right method
  - greatly simplifies the implementation of message sending in OO development.
- Allows different objects to respond appropriately to the same command, greatly reducing the complexity of the system.
  - E.g, in a payroll system, manager, office worker, and production worker objects all will respond to the *compute payroll* message -- but differently
- H_2O takes the forms…?
- Your behavior can be polymorphic when you approach a traffic light.
  - i.e., each subclass should be able to respond differently.

Adapted from: Keng Siau-University of Nebraska-Lincoln 44

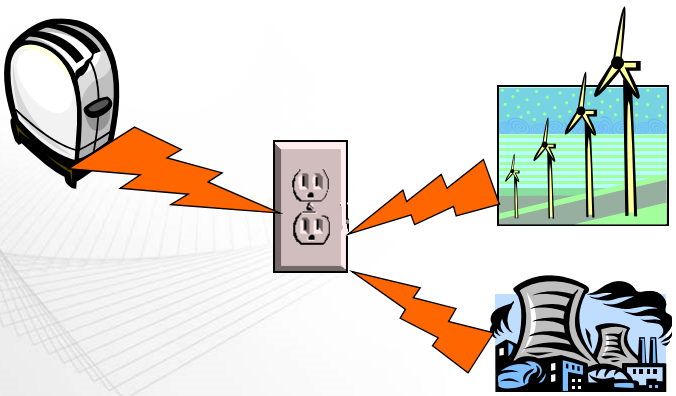
Objects **collaborate** with other objects To achieve some behavior



Source: Booch, 1994, 2007

Interface

- **Fundamental means of communication between objects**
- **Class design must specify the interfaces for proper instantiation and operation of objects**
- **Interface must describe how users of the class will interact with the class.**

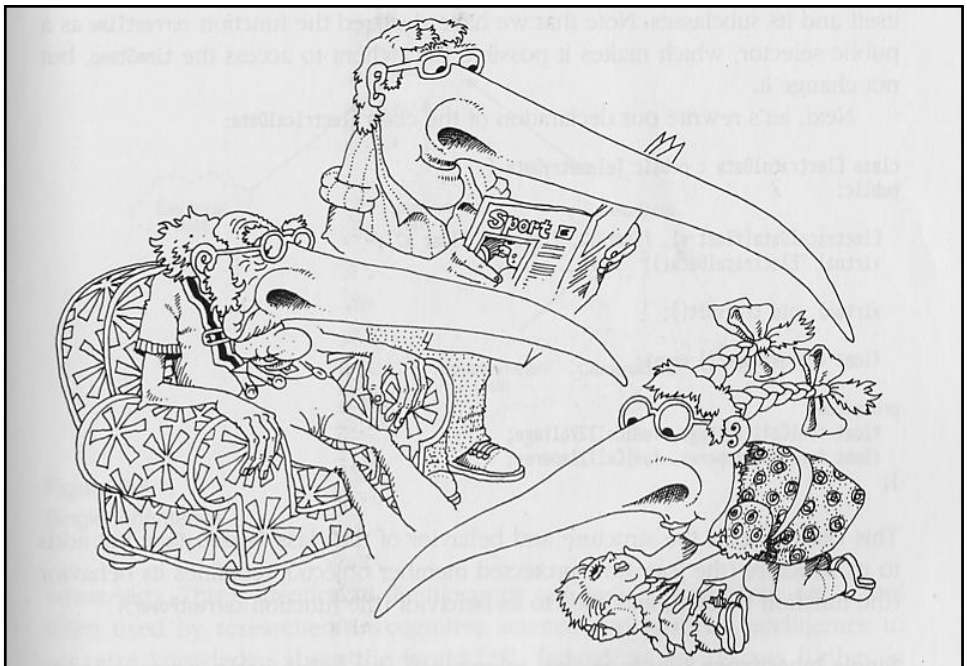


Source: Weisfeld,2000

Inheritance

- Inheritance allows classes to share and reuse behaviors and attributes
- A subclass inherits all of the properties and methods defined in its superclass
- The subclass has at least one attribute or method that differs from its superclass

Adapted from: Keng Siau-University of Nebraska-Lincoln



A subclass may inherit the structure and behavior of its superclass.

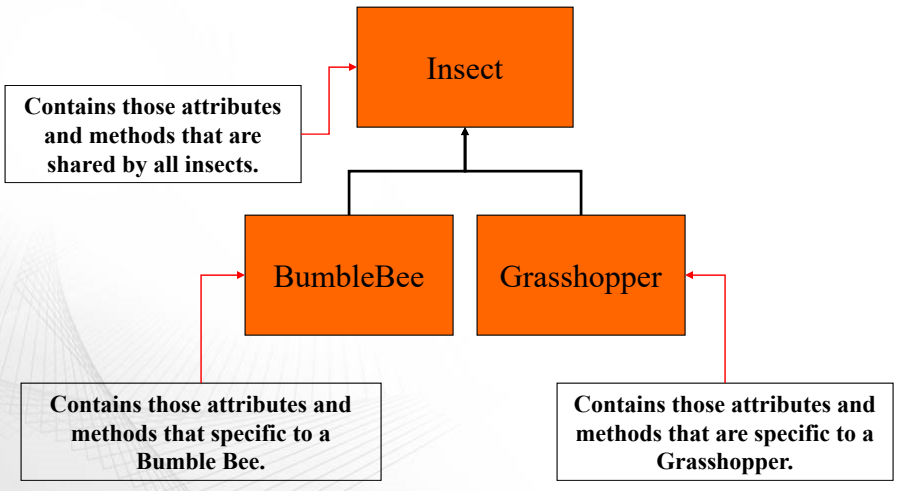
Source: Booch, 1994, 2007

What is Inheritance?
Generalization vs. Specialization

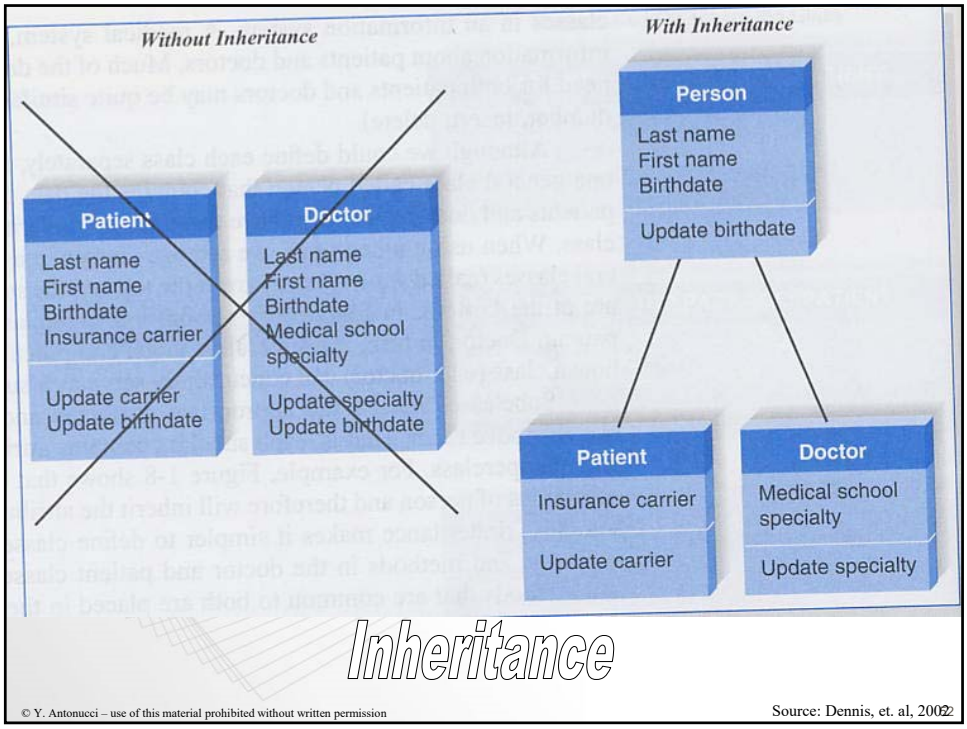
- Real-life objects are typically specialized versions of other more general objects.
- The term "insect" describes a very general type of creature with numerous characteristics.
- Grasshoppers and bumblebees are insects
  - They share the general characteristics of an insect.
  - However, they have special characteristics of their own.
    - grasshoppers have a jumping ability, and
    - bumblebees have a stinger.
- Grasshoppers and bumblebees are specialized versions of an insect.

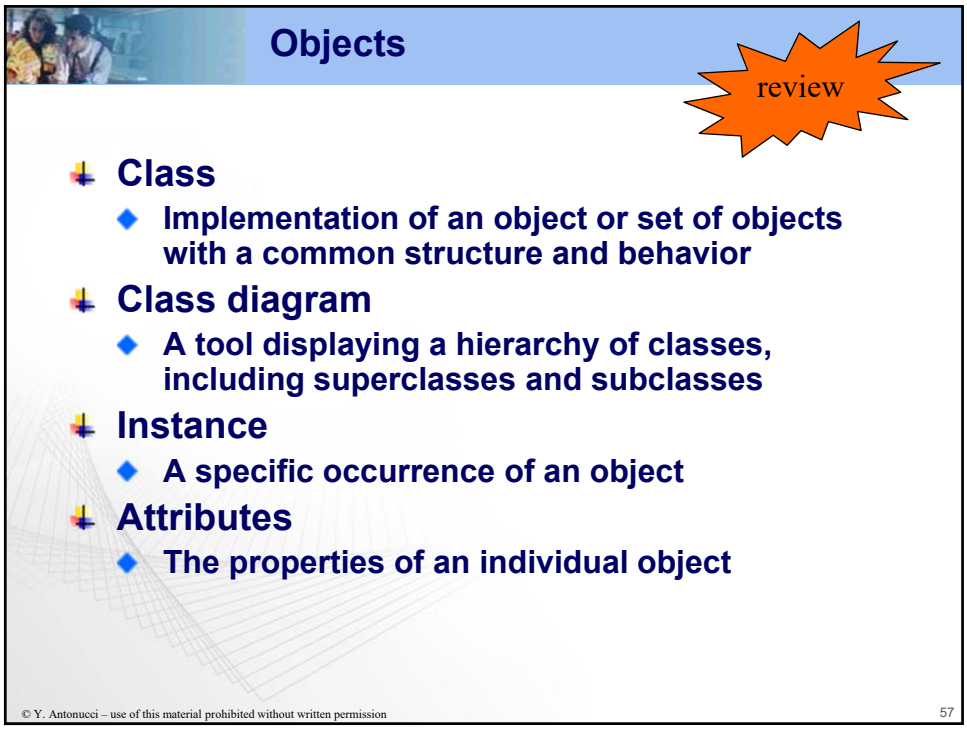
Source: Gaddis, 2013

Inheritance



Source: Gaddis, 2013


Without Inheritance
With Inheritance
Patient
Last name
First name
Birthdate
Insurance carrier
Update carrier
Update birthdate
Doctor
Last name
First name
Birthdate
Medical school specialty
Update specialty
Update birthdate
Person
Last name
First name
Birthdate
Update birthdate
Patient
Insurance carrier
Update carrier
Doctor
Medical school specialty
Update specialty
Inheritance
© Y. Antonucci – use of this material prohibited without written permission
Source: Dennis, et. al, 2002
22


Objects
review
Class
Implementation of an object or set of objects with a common structure and behavior
Class diagram
A tool displaying a hierarchy of classes, including superclasses and subclasses
Instance
A specific occurrence of an object
Attributes
The properties of an individual object
© Y. Antonucci – use of this material prohibited without written permission
57

Operations

review

- **Method**
  - **The code of an operation on the data of an object**
- **Event**
  - **The process of a trigger sending a message that results in an operation**

TERMINOLOGY REVIEW

review

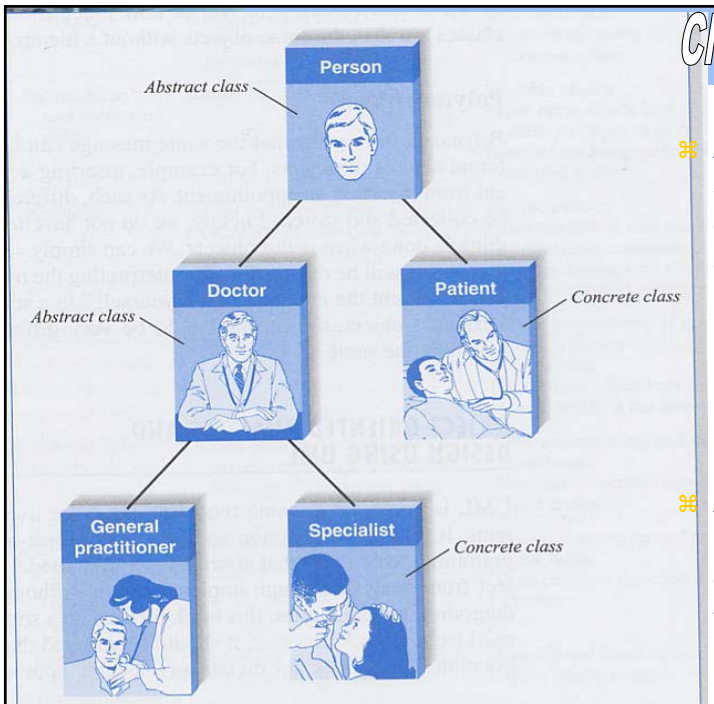
- **A mammal is an <u>object</u>**
- **Bear, buffalo, whale, and dolphin are <u>subtypes</u>**
- **Yogi, Wilbur, and Smokey are <u>instances</u> of the Bear subtype**
- **Eye color, ear size, and weight are <u>attributes</u>**

SUMMARY BY EXAMPLE

review

- A DVD player IS *an object*
- A Sony DVD player is *an object type*
- Serial #9234 of Sony DVDplayer is *an object instance*
- Playback, record, and audio dubbing are examples of DVD *operations*
- The concept that the DVD contains complex components you assume work is *encapsulation*
- When you use a remote control, you are sending *requests* to the DVD

Class Hierarchy



- An object-oriented system organizes classes into a subclass-superclass hierarchy
  - **superclass** = base class
  - **subclass** = derived class
- At the top of the class hierarchy are the most general classes and at the bottom are the most specific

Source: Dennis, et. al, 2002

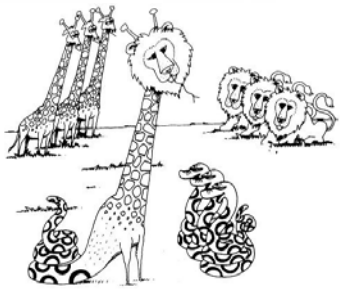
Dynamic Inheritance

- Allows objects to change and evolve over time
- Superclasses (or base classes) provide properties and attributes for objects
  - Changing superclasses changes the properties and attributes of a class
- Ability to add, delete, or change parents from objects (or classes) at run time

Adapted from: Keng Siau-University of Nebraska-Lincoln 93

Multiple Inheritance

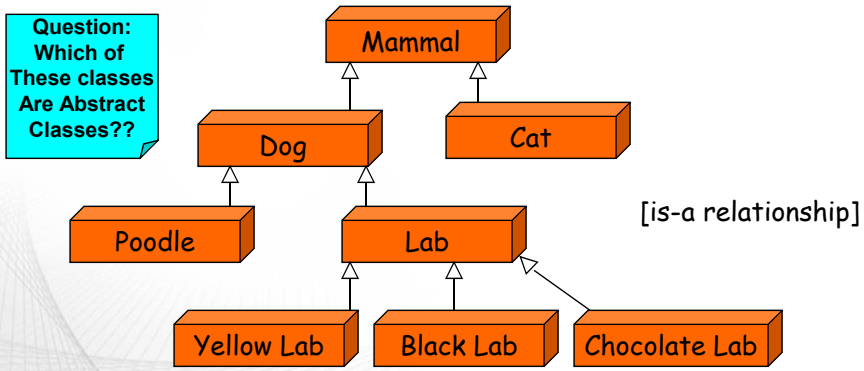
- A class can inherit its state (attributes) and behaviors from more than one superclass
- E.g., a utility vehicle inherits attributes from both the Car and Truck classes



Adapted from: Keng Siau-University of Nebraska-Lincoln 94

Abstraction

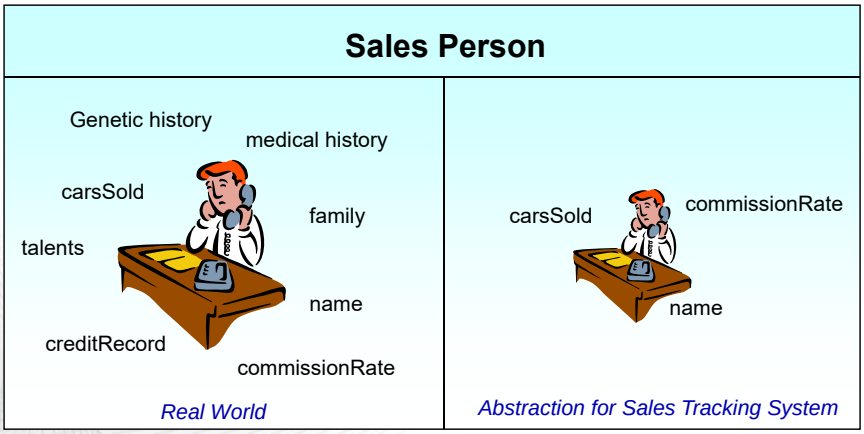
- A class inheritance tree can get very large …



- Therefore the power of inheritance lies in its abstraction!

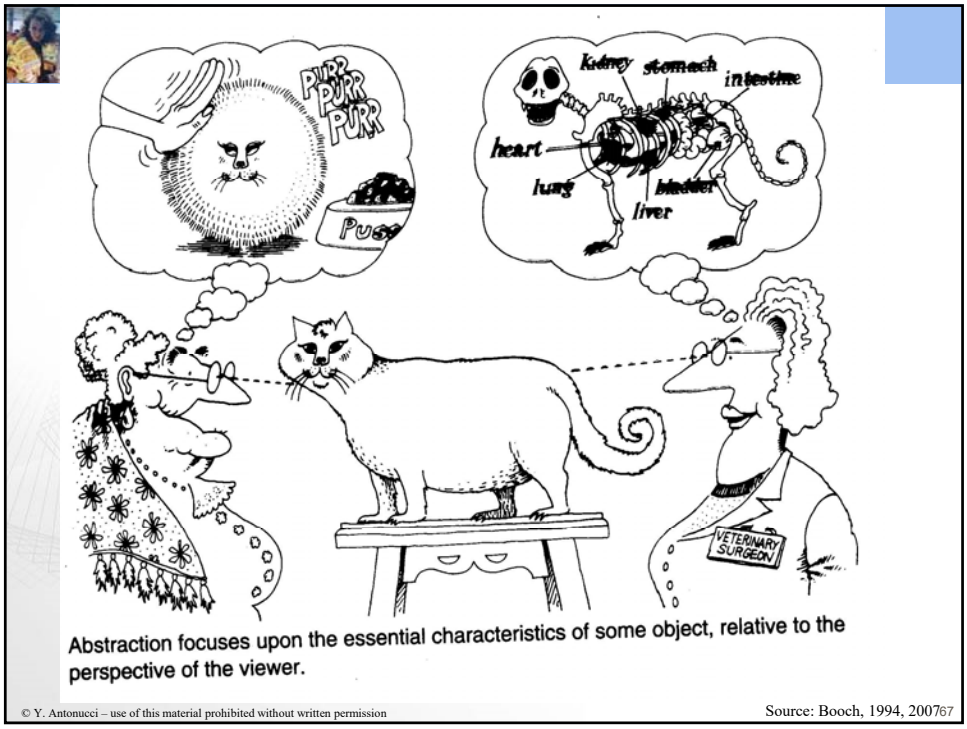
 65

Abstraction – only consider things Important to current situation. Ignore all else!



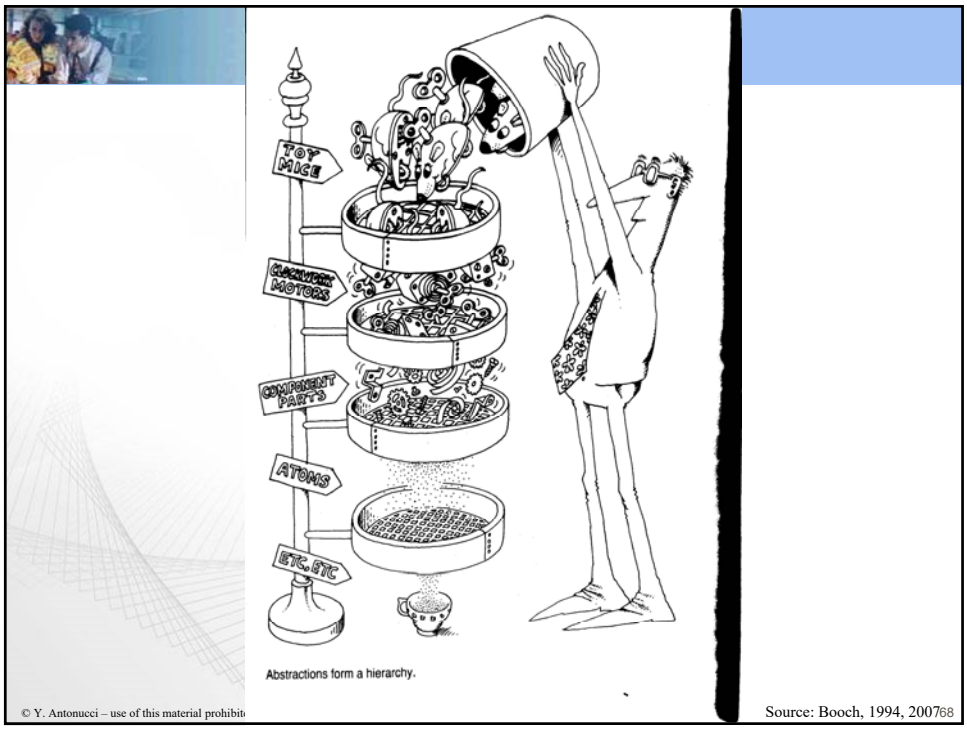
*Key in OO modeling: identify abstract classes whenever possible. Why do you think this is important?

 Source: Booch, 1994 66



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

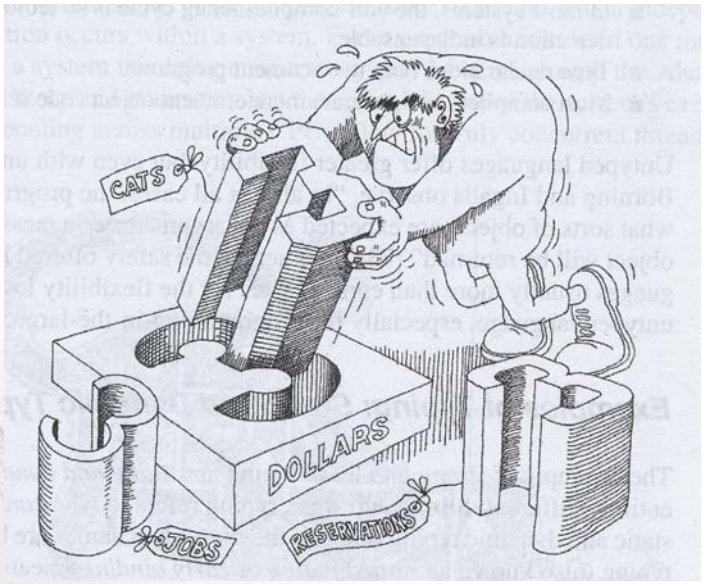
Source: Booch, 1994, 2007 67



Abstractions form a hierarchy.

Source: Booch, 1994, 2007 68

Strong Typing – prevents mixing of abstractions.



Source: Booch, 1994, 2007

Abstractions Depend on Use

- **DOT's abstraction of my car is quite different than the abstraction of my car that a game maker may want.**
  - **To a DOT data base system analyst**
    - **Attributes:**
      - Year/Make/Model, Color, License #, Owner, VID, …
    - **Protocol:**
      - Owner, Owner:, License, License:, etc.
  - **To a game software expert**
    - **Attributes:**
      - Display, Speed, Maneuverability, Condition,…
    - **Protocol:**
      - Show, Speed, Speed:, Condition, Crash, etc.
- **Note: The services (behaviors, responsibilities) of an object represent its contractual commitment or *protocol***

Composition (Aggregations)

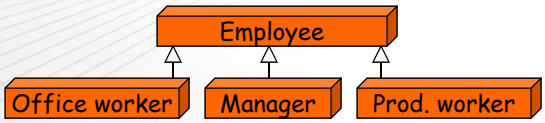
Objects can contain other objects!

- **All objects, except the most basic ones, are composed of and may contain other objects**
  - **A bicycle can contain wheels, handle bars, ….**
  - **a Unicycle can be constructed from most of the objects defined in a bicycle.**
  - **a car object is an aggregation of engine, seat, wheels, and other objects**
- **Has-a relationship**

 71

Another look at Polymorphism

- **Tightly coupled to inheritance…**
- **The same operation may behave differently on different classes – each class is able to respond differently to the same method!**
- **When a message is sent to an object, the object must have a method defined to respond to that message.**
  - **E.g, in a payroll system, manager, office worker, and production worker objects all will respond to the *compute payroll* message -- but differently**
  - **Even though Employee has a compute payroll method (generic for all company employees), a manager overrides this method to calculate payroll his/her own way (specific for managers). *Overriding – child replaces parent implementation***



 72

Modular Construction

Modularity packages abstractions into discrete units.

Source: Booch, 1994, 2007

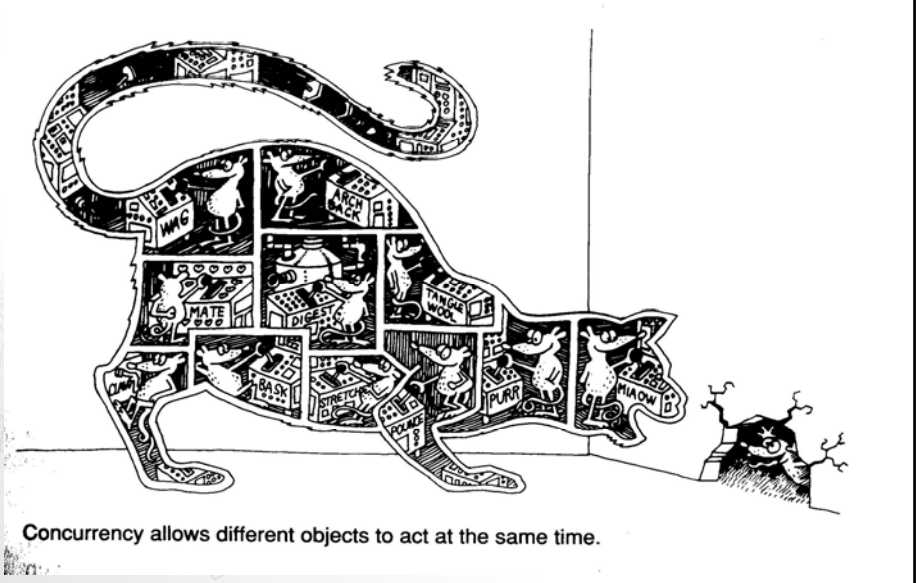
Object Relationships

- Association represents the relationship between objects and classes
- Associations are bidirectional
- Associations have cardinality
  - How many instances of one class may relate to a single instance of an associated class

More later…

Adapted from: Keng Siau-University of Nebraska-Lincoln

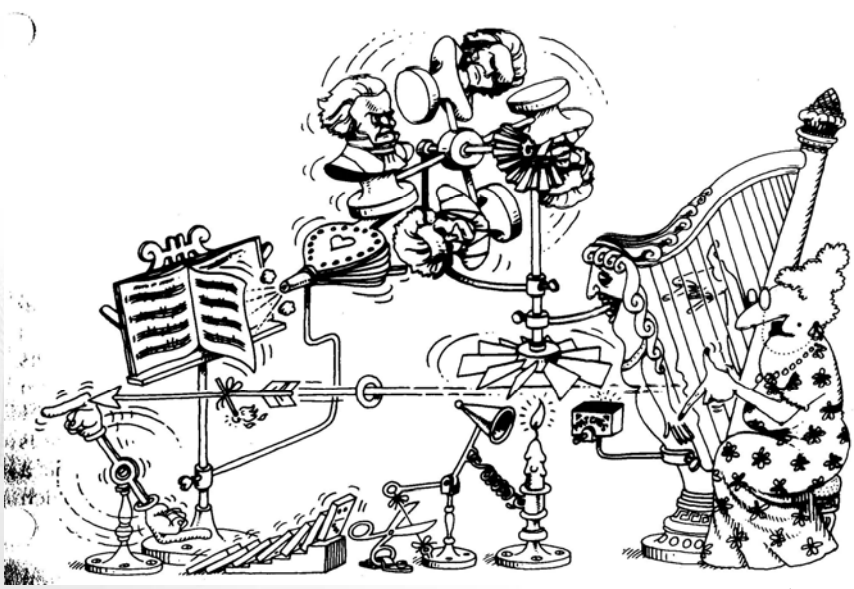
Concurrency



Concurrency allows different objects to act at the same time.

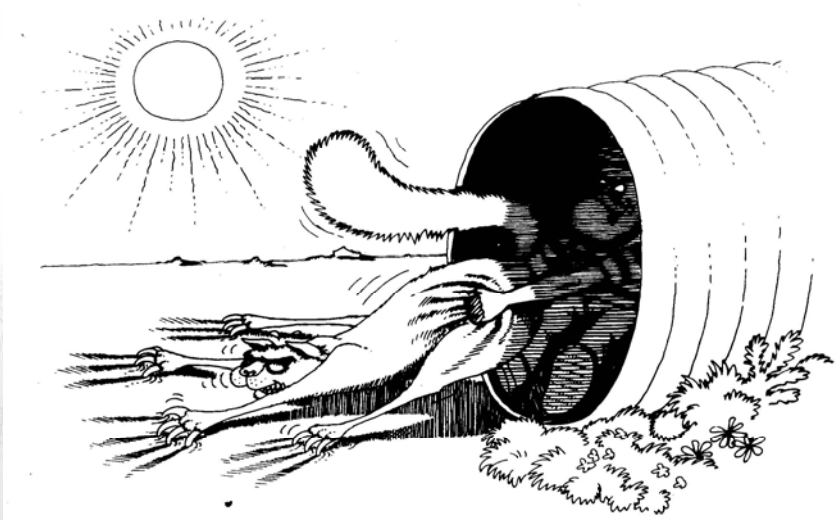
Source: Booch, 1994, 2007

Mechanisms – the means whereby objects collaborate to provide higher-level behavior.



Source: Booch, 1994, 2007

Persistence



Persistence saves the state and class of an object across time or space.

Source: Booch, 1994, 2007

Object-Oriented Programming and Design

- Object-oriented programming
  - Data and the code that operates on the data are packaged into a single unit called an object
- Object-oriented design
  - Identifies how objects interact with each other to solve a problem

class

method

```
public class Bowling
{
        public static void main(String args[]) throws IOException
        {
                String strRoll1 = "";
                String strRoll2 = "";
                String strRoll3 = "";
                int roll1 = 0;
                int roll2 = 0;
                int roll3 = 0;
                String status = "regular";
                String priorFrames = "regular";
                String lastFrame = "regular";
                int frameScore[] = new int[10];
                int runningTotal[] = {0,0,0,0,0,0,0,0,0,0};

                BufferedReader dataIn = new BufferedReader(new InputStreamReader(System.in));

                try{
```

event

Modeling Objects…

Object ID

Attributes

Methods

Invoice

Supplier

Related Order Amount

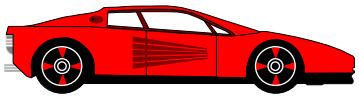
Post Invoice

Issue Payment

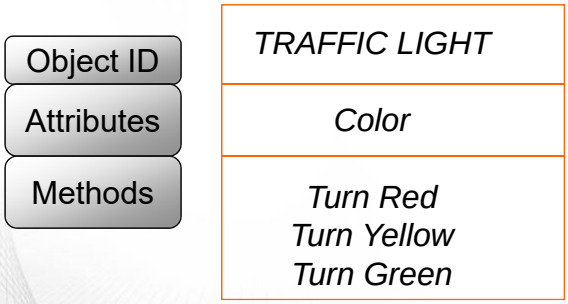
...

A MODEL . . .

- represents an aspect of reality
- helps us to understand reality
- assists us in inventing systems or redesigning business areas
- should be simpler than reality
  - for example, model cars are simpler than a real car



SAMPLE OBJECT STRUCTURE DIAGRAM

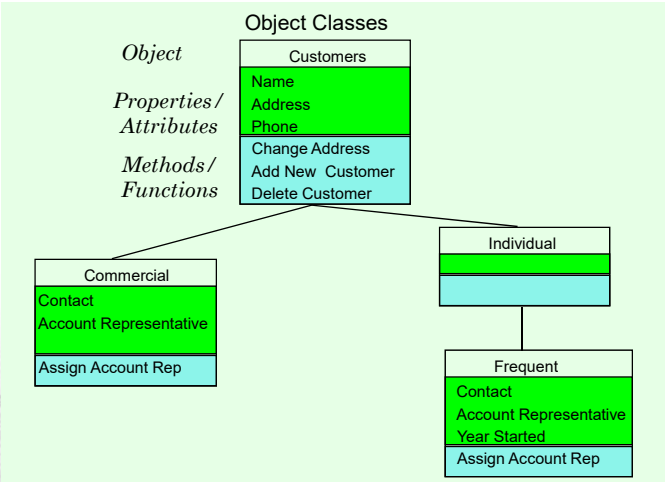


The operations are described in terms of what they do -- not how they do it.

YOU TRY ONE ...

- Draw an object structure diagram for the DVDplayer object
- Assume the attributes are buttons and on-screen messages
- Assume the methods are play, record, rewind, fast forward, and stop

Object Example



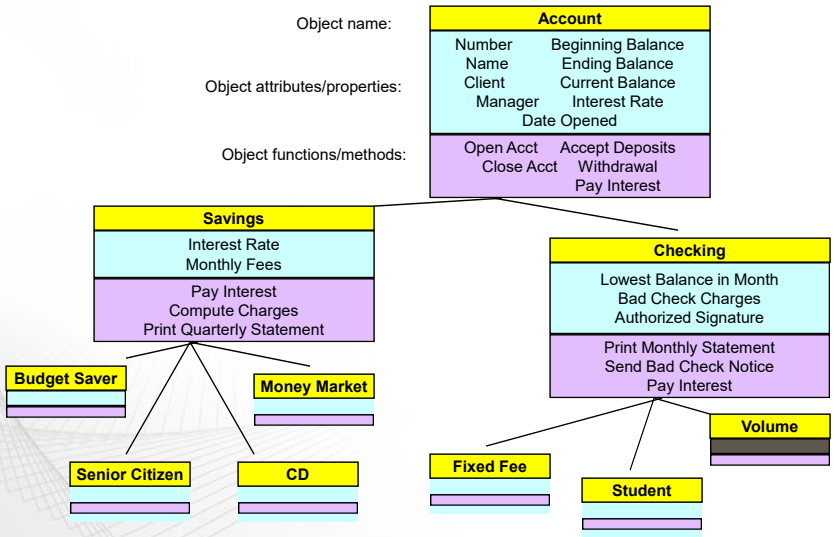
Source:Post, 2002

Remember…

Table 1-3 Ten Object-Oriented Programming and Design Concepts

1	An **object** is the basic unit of organization, a combination of a data element and a set of procedures.
2	A **method** is the code to perform a service or operation, including tasks such as performing calculations, storing values, and presenting results.
3	A **class** is an object or a set of objects that shares a common structure and a common behavior. A specific occurrence of an object class is called an **instance**.
4	A **subclass** is a lower-level category of a class with at least one unique **attribute** or method of its own.
5	A subclass **inherits** the attributes, methods, and variables from its superclass. A **superclass** is a higher-level category of class, from which the subclass inherits attributes, methods, and variables.
6	The tree-like structure showing the relationship of subclasses and superclasses is called a **generalization hierarchy** or class diagram.
7	A **message** requests objects to perform a method. A message is comprised of the object name together with the method.
8	An **event** occurs when a trigger causes an object to send a message.
9	**Encapsulation** is the process of hiding the implementation details of an object from its user by combining attributes and methods.
10	**Polymorphism** allows instructions to be given to objects in a generalized rather than specific, detailed command.

Object-Oriented Design



Source:Post, 2002

Rapid Application Development

- **Pre-built objects speed up program development**

Table 1-4 The Benefits of Object-Oriented Programming

BENEFIT	EXPLANATION
Reusability	The classes are designed so they can be reused in many systems, or modified classes can be created using inheritance.
Stability	The classes are designed for repeated use and become stable over time.
Easier design	The designer looks at each object as a black box and is not as concerned with the detail inside.
Faster design	The applications can be created from existing components.

Assignment #4 – due Feb. 28

NOTE: do not use any Examples presented!

- **You are to come up with an example showing the following:**
  - **Draw the associations – show the class structure of objects in your example** [similar to diagram in Abstraction slide]
    - **Indicate which classes are abstract in your diagram**
  - **Draw an object structure diagram for one of the objects**
  - **Select one class and briefly explain what the common attributes are.**
  - **Briefly explain one example for each of the following using your example:**
    - **A method**
    - **A message**
    - **Encapsulation**
    - **An interface**
    - **polymorphism**