Object-Oriented Design

IT4GIS
Keith T. Weber, GISP
GIS Director
ISU-GIS Training and Research Center

Topics

• During the balance of this semester, we will pursue and follow two learning threads
  – Object-relational databases
  – The Geo-Web
• These two threads are interwoven

To understand Object-Relational Databases…

• We need to understand both relational concepts and
  Object-oriented concepts (this week)
The Early Days…

• Computer programming from the caveman era

Why…Object-Oriented

• A brief history of computer programming…

The concept of encapsulation!

Today’s Goals

• We will dissect “Object-Oriented” to learn what it really is and how it relates to object-relational databases
  – What is a class?
  – What is an object?
• Enable you to identify inheritance, aggregation, and dependency relationships between classes
• Understand class attributes and object properties
• Become familiar with new terminology
What is a CLASS?

• A *class* is a computer construct representing a concept bound in a cohesive package
  – Some are concrete (i.e., real world)
    • Bank account
    • Rental item
    • Database item
    • Pile
  – Others are abstract
    • Scanner
    • Stream
    • Math

Discovering CLASSES

• Simple Rule:
  – Look for *nouns* in descriptions
  – Obviously not all nouns are classes
  – But at least this approach can allow one to create a list of *candidate classes*

What is an OBJECT

• An *instance* of a CLASS
• That contains meaningful data
• OBJECTS occupy memory space at runtime
  – If not, they are CLASSES
  – For example: data type vs. double
A little Quiz…

• #1 Class or Object?

Dog

Scooby-Doo

D o g is a generalization of Scooby-Doo

A little Quiz (cont’d)…

• #2 Class or Object?

Dog

Animal

The concept of subclass!

Dog is a subclass of the Animal class

Scooby-Doo

Animal is a generalization of Dog

A little Quiz (cont’d)…

• #3 Class or Object?

Animal

Bird

Dog

The concept of polymorphism!
Key Points

- Many classes already exist and are at our disposal when we design a database.
- We use inheritance to add capabilities to our projects.
  - A subclass inherits from its superclass.
  - i.e., a child inherits from its parent.

Inheritance in the Geodatabase
Defining a CLASS

• After a class has been identified we can define:
  – The behavior of each class
    • Methods (verbs)
  – And the attributes of each class

BEHAVIOR

Relationships Between CLASSES

• We have learned about inheritance as one (1) relationship between classes

  There are three (3) important relationships
    • Inheritance
    • Aggregation
    • Dependency
1- Inheritance

- Is-a relationship
- Relationship between a more general class (superclass) and a more specialized class (subclass)
- Every
  - savings account is a bank account
  - DVD rental is a rental

2- Aggregation

- Has-a relationship
  - Each Dog has a Paw (dog is not a generalization of paw!)
- One class (Appendages) contains references to another class (Dog)

Inheritance vs. Aggregation

- Often confused
- Questions?
Example

- Car is a Vehicle – Inheritance
- Car has a set of Tires – Aggregation

Figure 6
UML Notation for Inheritance and Aggregation

3- Dependency

- Dependency occurs when a class uses or relies on another class
- This is a Uses relationship
  - Example: an application may depend on the Scanner class to read input

Class Diagram

[Diagram showing BankAccount class with attributes and methods]

What type of Method behaviors are these?
Attributes help define a given class and instantiate it into an object.

**Class Diagram**

Attributes help define a given class and instantiate it into an object.

**Dog example**

- Name of the class = Dog
- Methods:
  - Example of inheritance
  - Example of aggregation
  - Example of dependence
- Attributes?
Instantiate into an object

- Three features characterize objects:
  - **Identity**: specific attribute (property) settings have been made for the class. This distinguishes it from all other objects.
  - **State**: Describes the data stored in the object. *WHERE DID THIS COME FROM?*
  - **Behavior**: describes the method in the object’s interface through which the object can be used (how do we make the dog bark?)

Instantiating the Dog CLASS

- CLASS (DOG)
- Attributes (Properties)
  - NAME = Scooby-Doo
  - HEIGHT = 36
  - WEIGHT = 145
- Methods
  - [Uses] bark- “Rooby roo”
  - etc.

Key Concepts

- Understand the difference between a CLASS and an OBJECT
- Understand new terms:
  - Encapsulation, polymorphism, superclass, subclass, behavior, attributes, instantiation
- Understand --and be able to differentiate-- the three types of behaviors