

Object Oriented Design Essentials

Object Interaction

Object Interaction

Chapter 2

Objectives

You will learn:

- How objects interact.
- Understand the relationship between objects.
- Concepts of actors, servers, and agents.
- Understand inheritance, aggregation, association, and instantiation.

Object Oriented Design Essentials

Object Interaction

Object Interaction

- The objects in a system are not independent; they interact with each other.
- Deciding on the classes, with their attributes and operations, is insufficient for design purposes.
- It will also be necessary to identify how different classes interact with, or are related to, each other.
- This can be done in 2 ways:
 - by *communicating* with each other.
 - by having some sort of *relationship*.

SYS-ED/Computer Education Techniques, Inc.

3

Communicating Objects

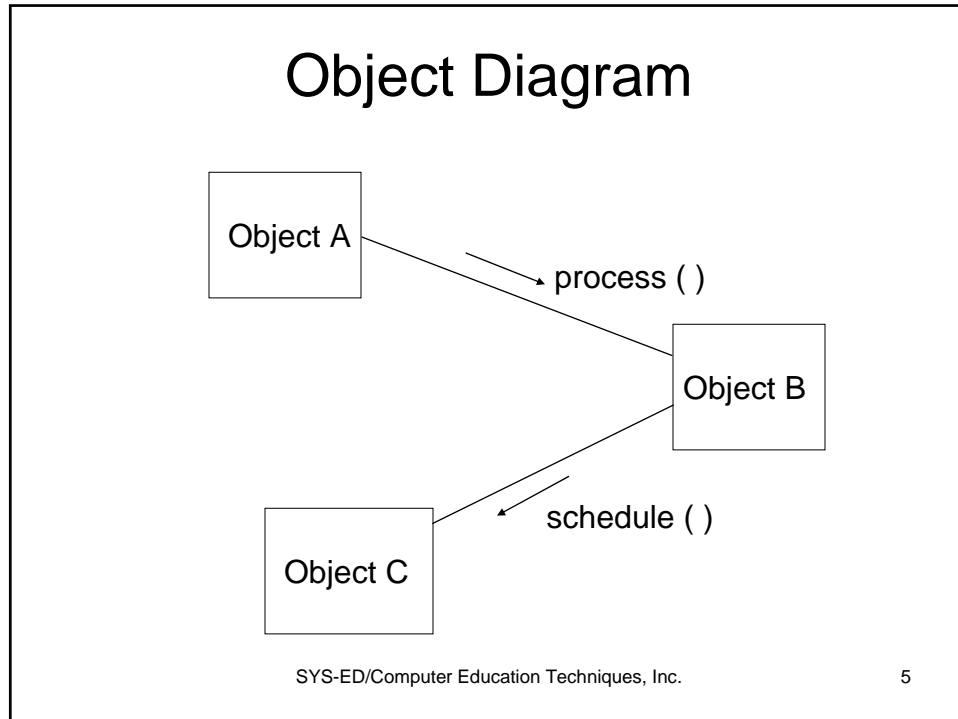
- Objects communicate with each other via message passing.
- To send a message from object A to object B it will be necessary to invoke one of object B's operations from object A.
- The message may include information being sent to object A, and/or it may bring back a reply or return value.
- Design documentation should incorporate message passing between objects.
- Object diagrams should be drawn showing such communicational links between the different objects of the system.

SYS-ED/Computer Education Techniques, Inc.

4

Object Oriented Design Essentials

Object Interaction



Actors, Servers, and Agents

- Actors, servers and agents are very simple concepts, and are very useful when drawing up object diagrams.
- Actors send messages to but never receive messages from other objects.
- Servers never send messages to, only receive messages from other objects.
- Agents send and receive messages.

Object Oriented Design Essentials

Object Interaction

Class Relationships

- There are 4 main relationships:
 - » inheritance
 - » aggregation
 - » association
 - » instantiation
- The cardinality of the relationships aggregation and association may be one-to-one, one-to many, or many-to-many.

SYS-ED/Computer Education Techniques, Inc.

7

Inheritance

- A generalization / specialization relationship between classes. Useful phrase - "is a type of" or "is a".
- e.g. Part-time student is a student.
- A Part-time student is a specialized subclass of the more general class student.
- Part-time student inherits all the attributes and operations of the superclass student.
- Classes may be formed into a hierarchy.

SYS-ED/Computer Education Techniques, Inc.

8

Object Oriented Design Essentials

Object Interaction

Inheritance - Advanced Concepts

Multiple inheritance

- a class may inherit from more than one superclass.
- e.g. hovercraft, inheriting from boat and plane.

Polymorphism (many forms)

- a subclass may alter inherited attributes or operations.
- e.g. *line graph* and *bar graph* both inherit from *graph* which has an operation *draw*, but both have their own variations of the operation.

Abstract classes

- have no instances e.g. vehicle.
- usually towards the top of an inheritance hierarchy.

SYS-ED/Computer Education Techniques, Inc.

9

Aggregation

- A whole/part relationship between classes.
- Useful phrase “part of”, and sometimes “has”
e.g. a page is a part of a book.
- Also known as container/component relationship.
- Cardinality:
 - one-to-one, or one-to-many.

SYS-ED/Computer Education Techniques, Inc.

10

Object Oriented Design Essentials

Object Interaction

Association

- Less clear-cut than inheritance and aggregation.
- Denotes some semantic dependency between the classes (other than “is a type of” or “is a part of” or “has”).
- Often when one class is associated with another then one has a *role* to play with respect to the other.
- Association is also known as *utilization* or *use*, with one class acting as a server and the other a client.
- Often an association is changed into another more clear-cut relationship as a design is developed.

SYS-ED/Computer Education Techniques, Inc.

11

Instantiation

- Instantiation is unlike the previous relationships in that it is not a relationship between classes.
- Instantiation is a relationship between a class and an object.
- An object is an instance of a class.
- At any one time a class may have zero or more object instances.
- Classes can be thought of as *static* entities, with fixed properties: attributes, operations, a name, an interface.
- Conversely, objects may be thought of as *dynamic* entities with changing properties - the values of their attributes, the status of execution of their operations.

SYS-ED/Computer Education Techniques, Inc.

12