CLASS DIAGRAM

Object Oriented Analysis and Design

Using the UML

Introduction to Object Orientation

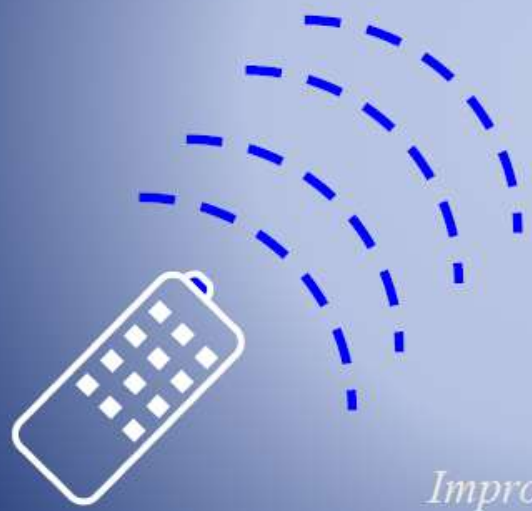
Abstraction

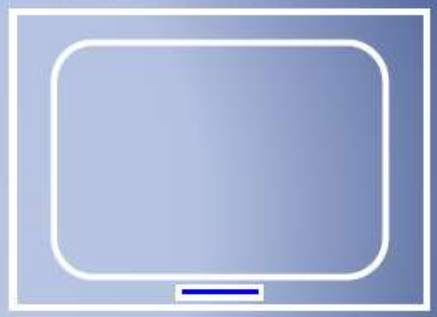
Abstraction is another good feature of OOAD. Abstraction means to show only the necessary details to the client of the object. Do you know the inner details of the Monitor of your PC? What happen when you switch ON Monitor? Does this matter to you what is happening inside the Monitor? No Right, Important thing for you is weather Monitor is ON or NOT. When you change the gear of your vehicle are you really concern about the inner details of your vehicle engine? No but what matter to you is that Gear must get changed that's it!! This is abstraction; show only the details which matter to the user.


What is Abstraction?
Salesperson
Not saying Which salesperson – just a salesperson in general!!!
Customer
Product
Manages Complexity

What is Encapsulation?

- Hide implementation from clients
  - Clients depend on interface





How does an object encapsulate?
What does it encapsulate?

Improves Resiliency

What is Modularity?

- The breaking up of something complex into manageable pieces

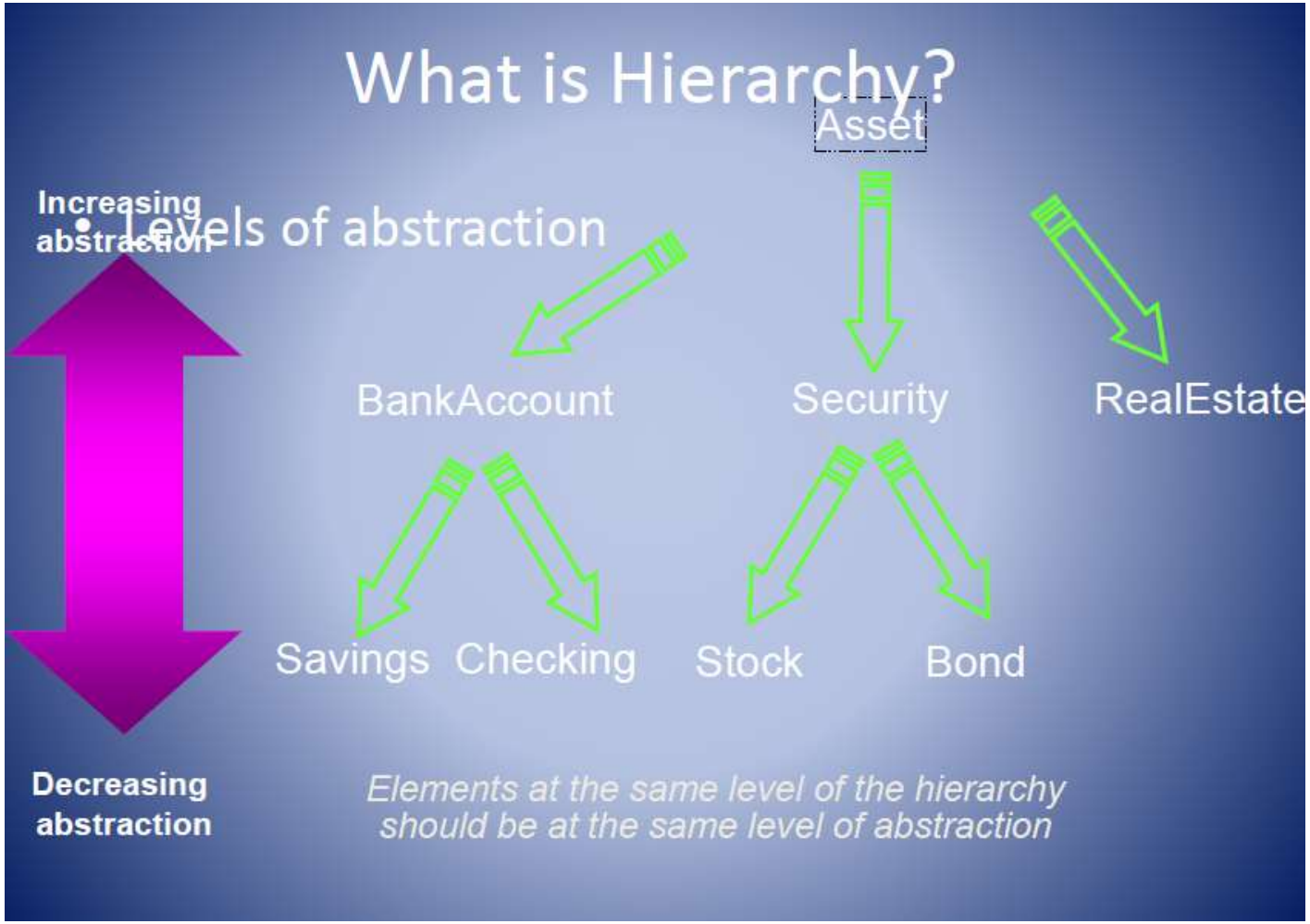
Order Entry

Order Processing System

Order Fulfillment

Billing

Manages Complexity


What is Hierarchy?
Asset
Levels of abstraction
Increasing abstraction
BankAccount
Security
RealEstate
Savings
Checking
Stock
Bond
Decreasing abstraction
Elements at the same level of the hierarchy should be at the same level of abstraction

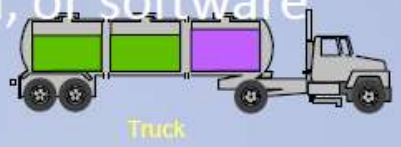
PM tutor
Empowering Excellence

Basic Concepts of Object Orientation

- Object
- Class
- Attribute
- Operation
- Relationships

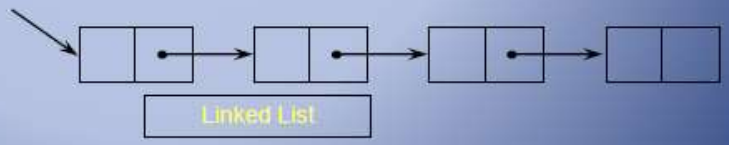
What is an Object?

- Informally, an object represents an entity, either physical, conceptual, or software
  - Physical entity



  - Conceptual entity

  - Software entity



A More Formal Definition

- An object is a concept, abstraction, or thing with sharp boundaries and meaning for an application
- An object is something that has:
  - State
  - Behavior
  - Identity

Representing Objects

- An object is represented as rectangles with underlined names

: Professor

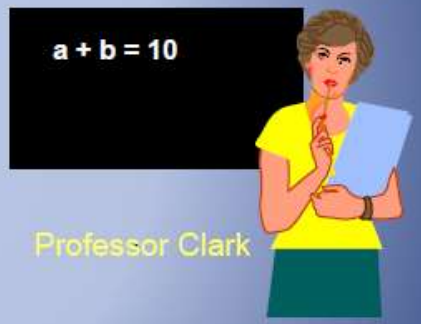
Class Name Only

ProfessorClark

Object Name Only

ProfessorClark : Professor

Class and Object Name



(stay tuned for classes)

What is a Class?

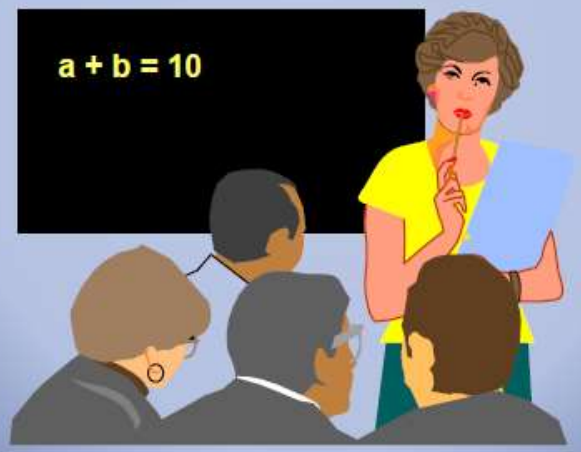
- A class is a description of a group of objects with common properties (attributes), behavior (operations), relationships, and semantics
  - An object is an instance of a class
- A class is an abstraction in that it:
  - Emphasizes relevant characteristics
  - Suppresses other characteristics

OO Principle: Abstraction

Sample Class

Class
Course

Properties
Name
Location
Days offered
Credit hours
Start time
End time



Behavior
Add a student
Delete a student
Get course roster
Determine if it is full

Representing Classes

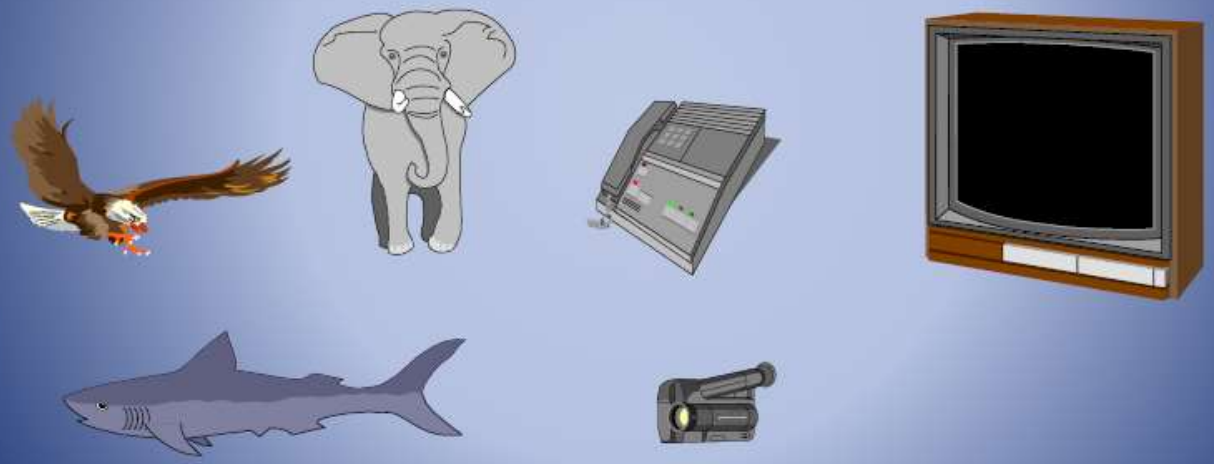
- A class is represented using a compartmented rectangle

Class Compartments

- A class is comprised of three sections
  - The first section contains the class name
  - The second section shows the structure (attributes)
  - The third section shows the behavior (operations)

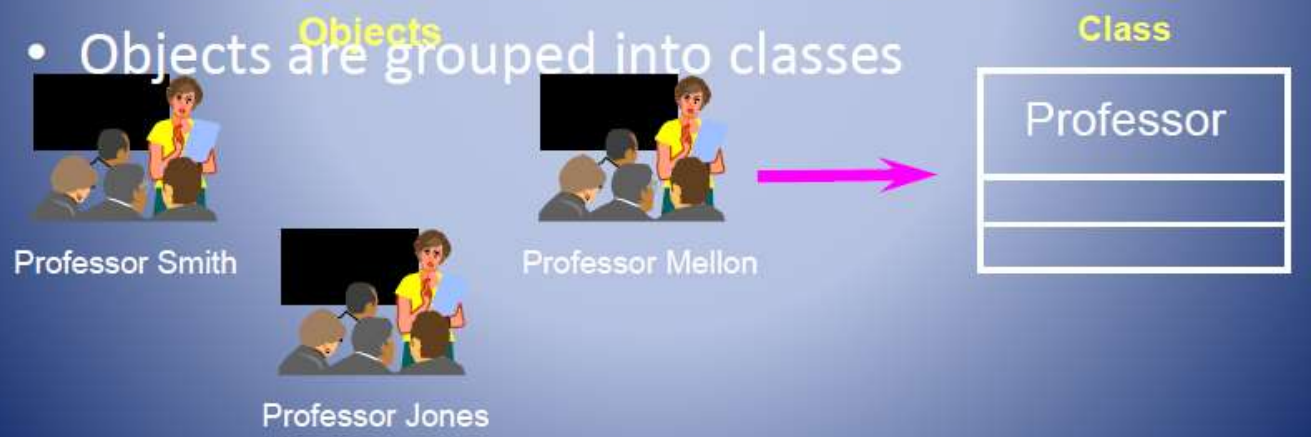
Classes of Objects

- How many classes do you see?



The Relationship Between Classes and Objects

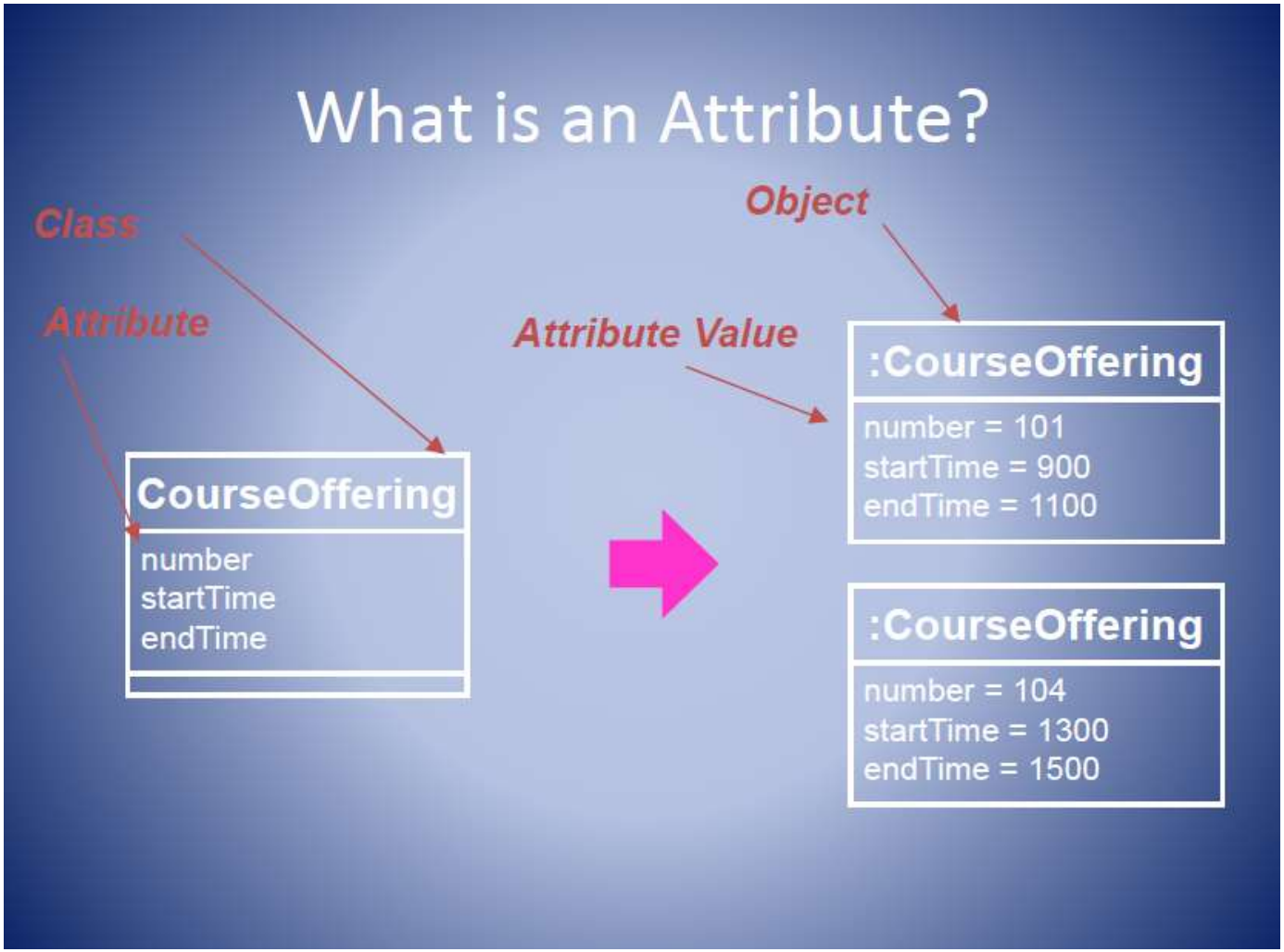
- A class is an abstract definition of an object
  - It defines the structure and behavior of each object in the class
  - It serves as a template for creating objects
- Objects are grouped into classes

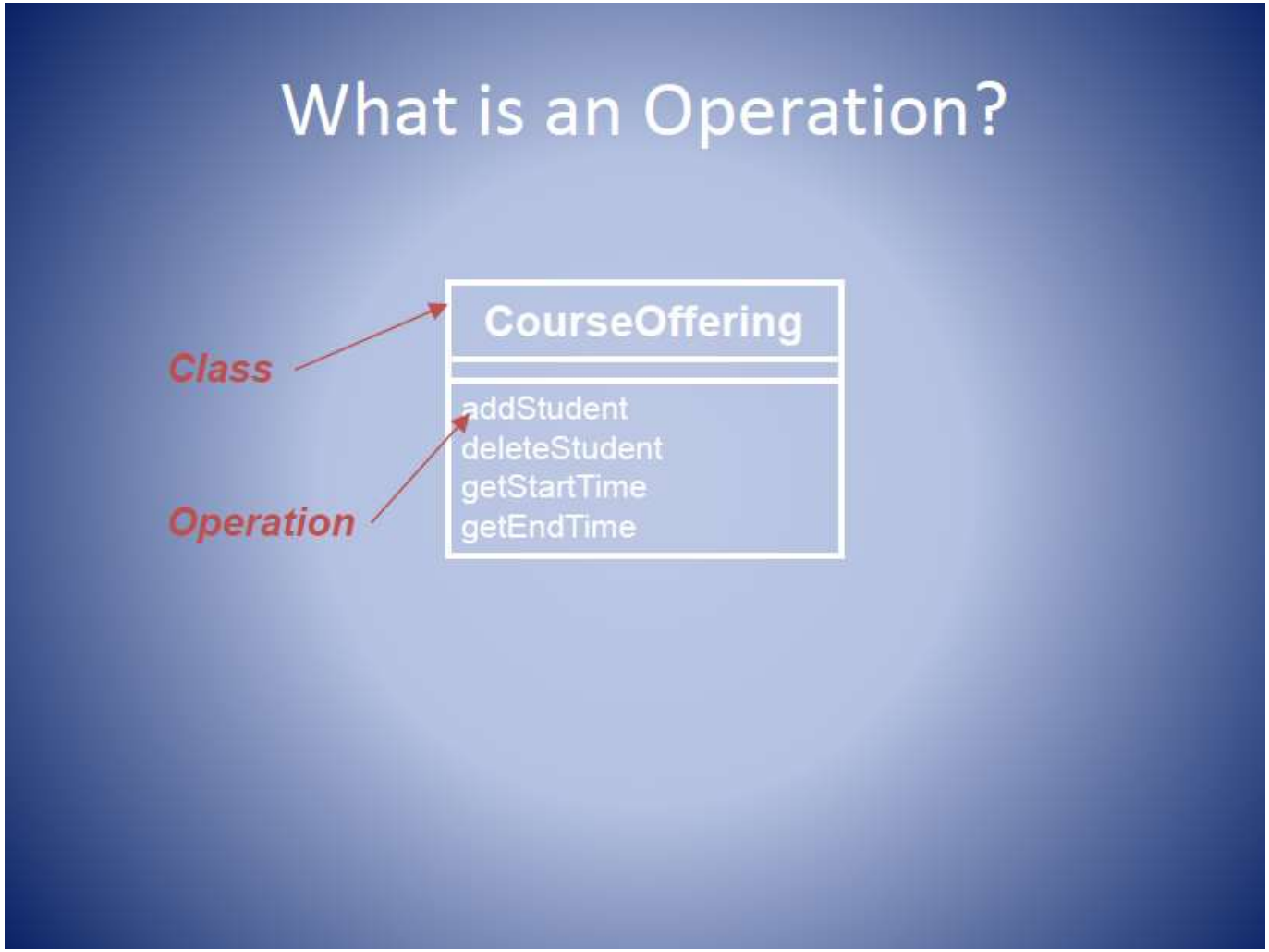


PM tutor
Empowering Excellence

Basic Concepts of Object Orientation

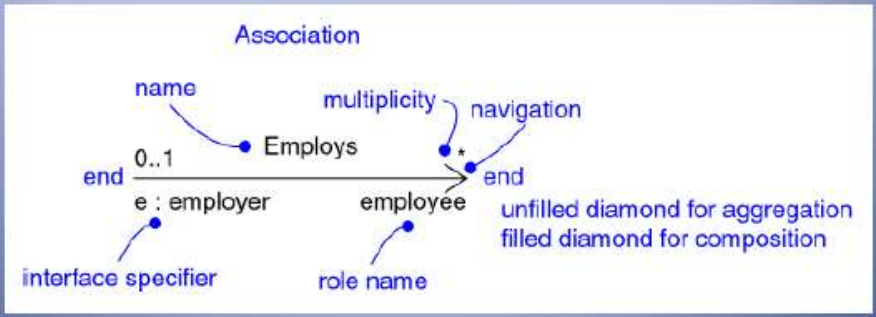
- Object
- Class
- Attribute
- Operation
- Interface (Polymorphism)
- Component
- Package
- Subsystem
- Relationships


What is an Attribute?
Class
Attribute
Object
Attribute Value
CourseOffering
number
startTime
endTime
:CourseOffering
number = 101
startTime = 900
endTime = 1100
:CourseOffering
number = 104
startTime = 1300
endTime = 1500


What is an Operation?
Class
Operation
CourseOffering
addStudent
deleteStudent
getStartTime
getEndTime

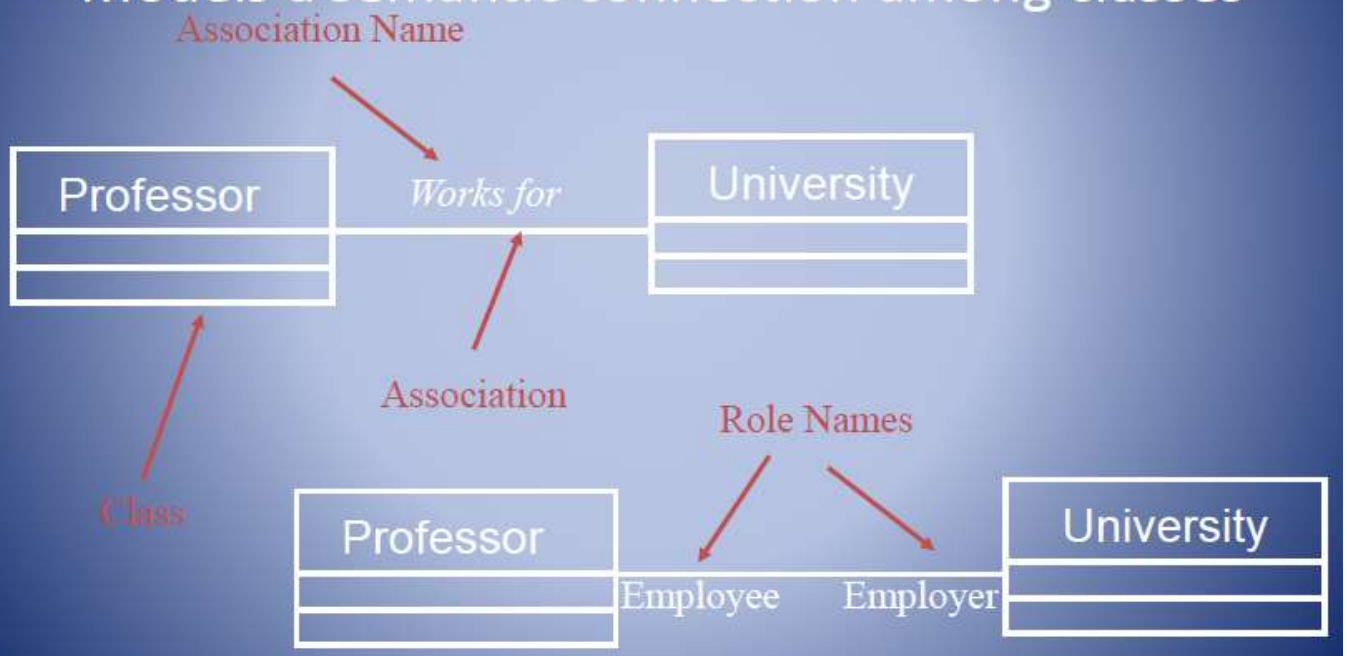
Relationships

- Association
  - Aggregation
  - Composition
- Dependency
- Generalization
- Realization



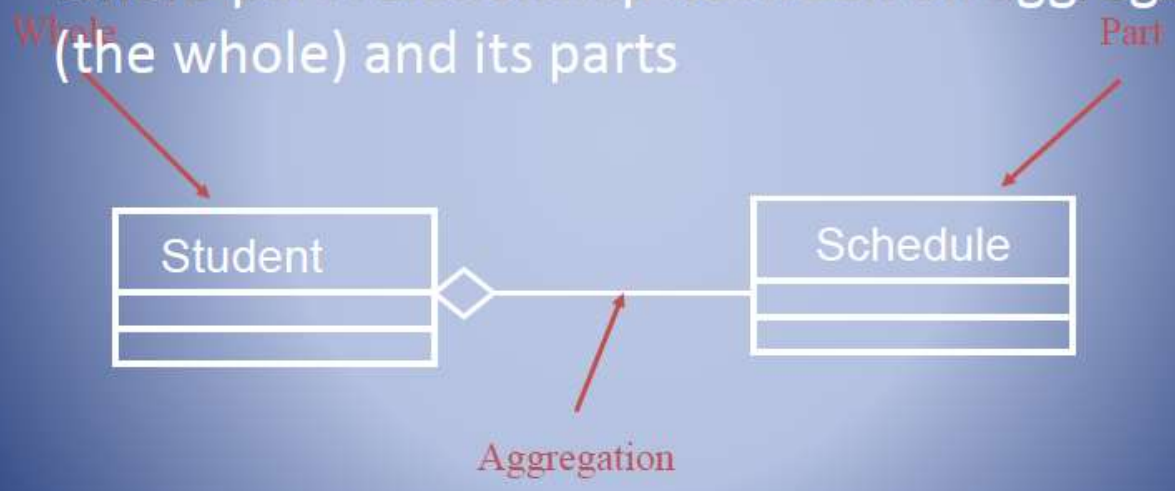
Relationships: Association

- Models a semantic connection among classes



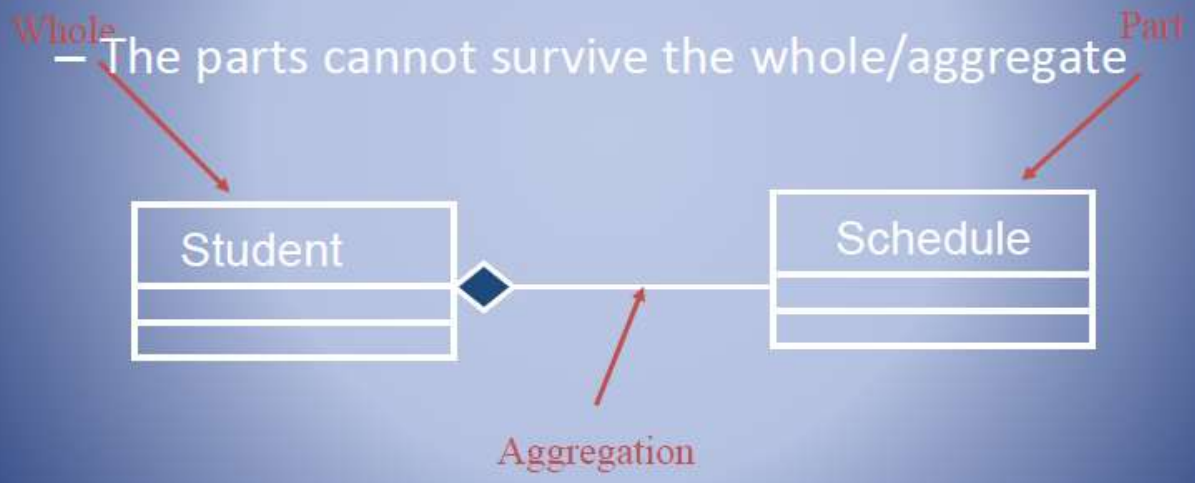
Relationships: Aggregation

- A special form of association that models a whole-part relationship between an aggregate (the whole) and its parts



Relationships: Composition

- A form of aggregation with strong ownership and coincident lifetimes
  - The parts cannot survive the whole/aggregate

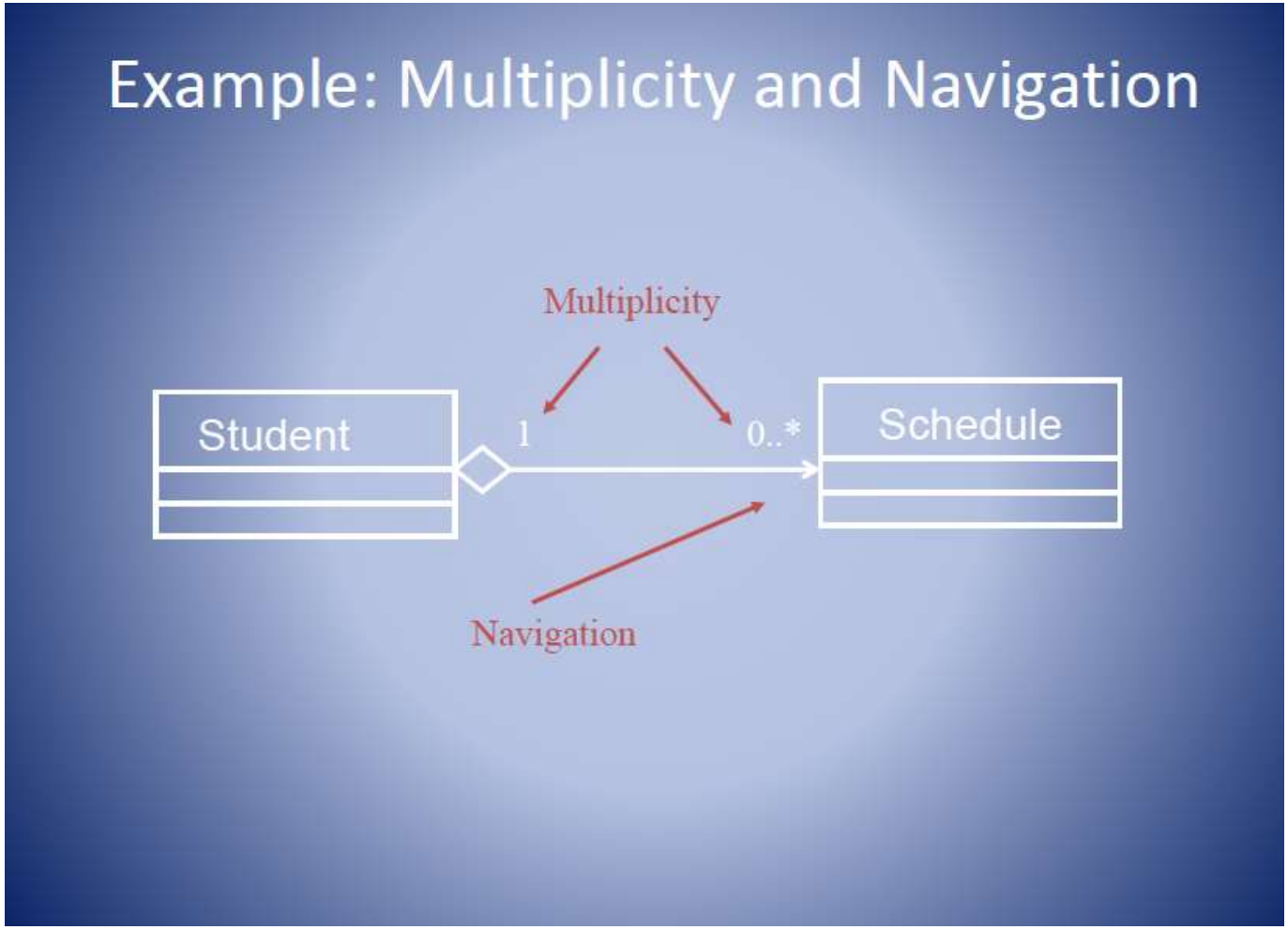


Association: Multiplicity and Navigation

- Multiplicity defines how many objects participate in a relationships
  - The number of instances of one class related to ONE instance of the other class
  - Specified for each end of the association
- Associations and aggregations are bi-directional by default, but it is often desirable to restrict navigation to one direction
  - If navigation is restricted, an arrowhead is added to indicate the direction of the navigation

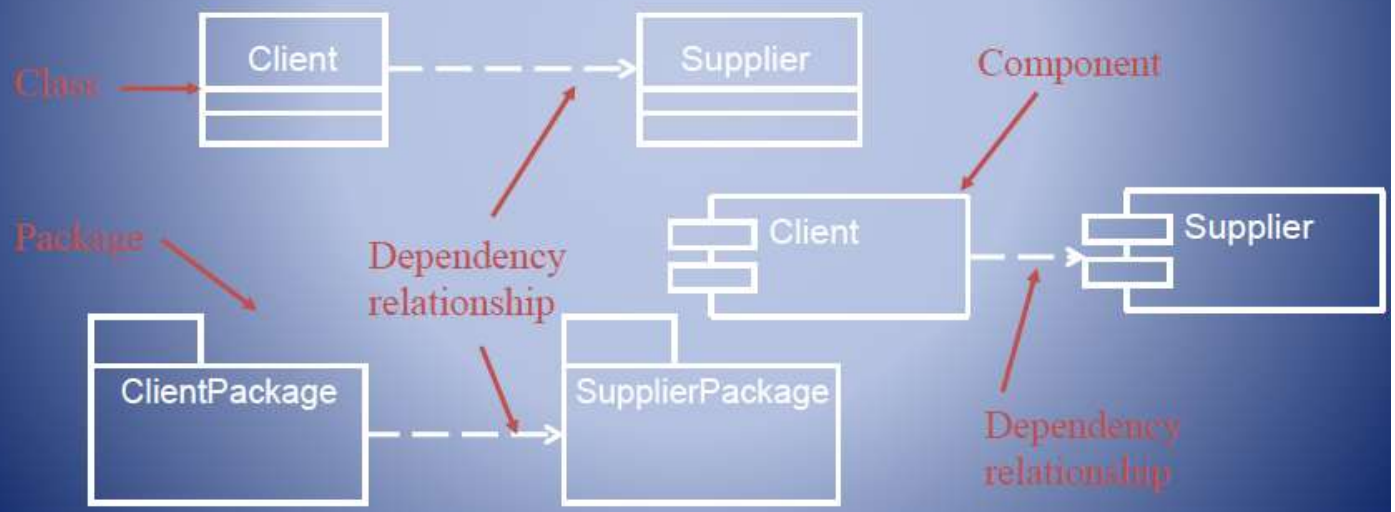
Association: Multiplicity

Unspecified	
Exactly one	1
Zero or more (many, unlimited)	0..*
	*
	1..*
One or more	0..1
Zero or one	2..4
Specified range	2, 4..6
Multiple, disjoint ranges	


Example: Multiplicity and Navigation
Multiplicity
Student
1
0..*
Schedule
Navigation

Relationships: Dependency

- A relationship between two model elements where a change in one **may** cause a change in the other
- Non-structural, "using" relationship

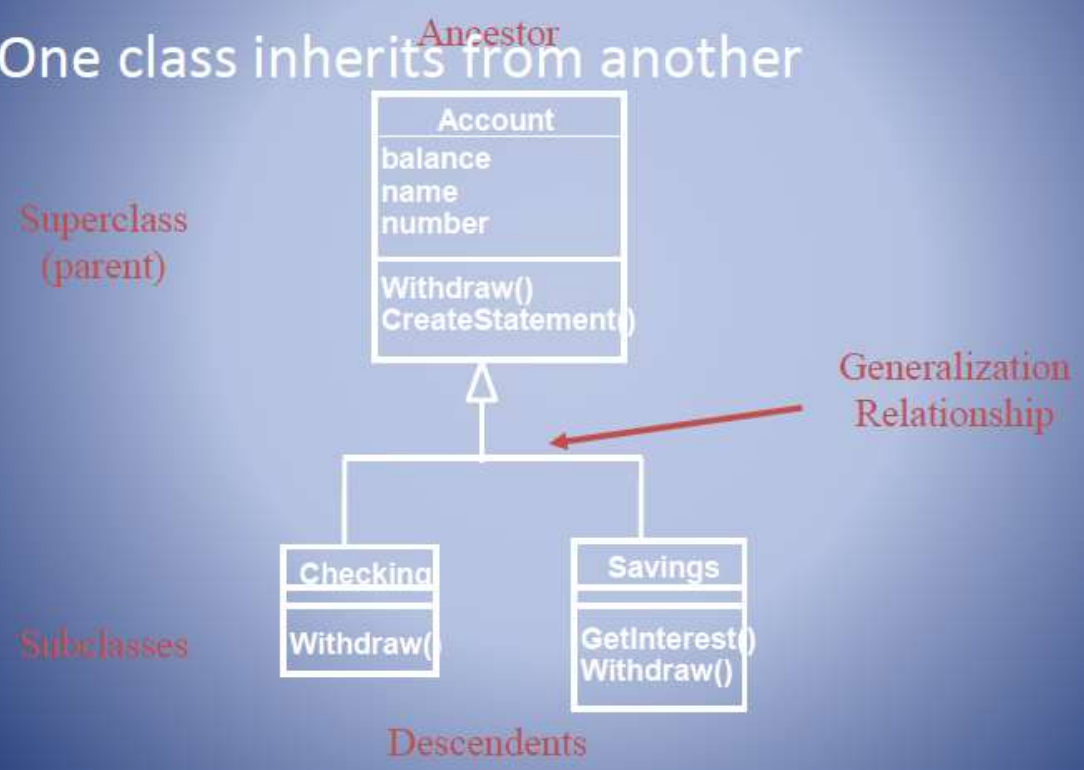


Relationships: Generalization

- A relationship among classes where one class shares the structure and/or behavior of one or more classes
- Defines a hierarchy of abstractions in which a subclass inherits from one or more superclasses
  - Single inheritance
  - Multiple inheritance
- Generalization is an "is-a-kind of" relationship

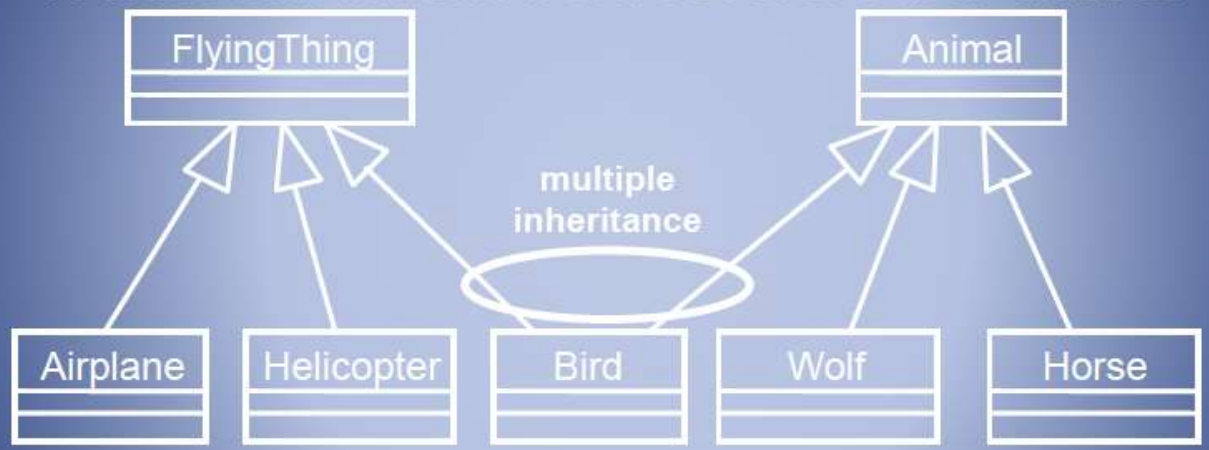
Example: Single Inheritance

- One class inherits from another



Example: Multiple Inheritance

- A class can inherit from several other classes



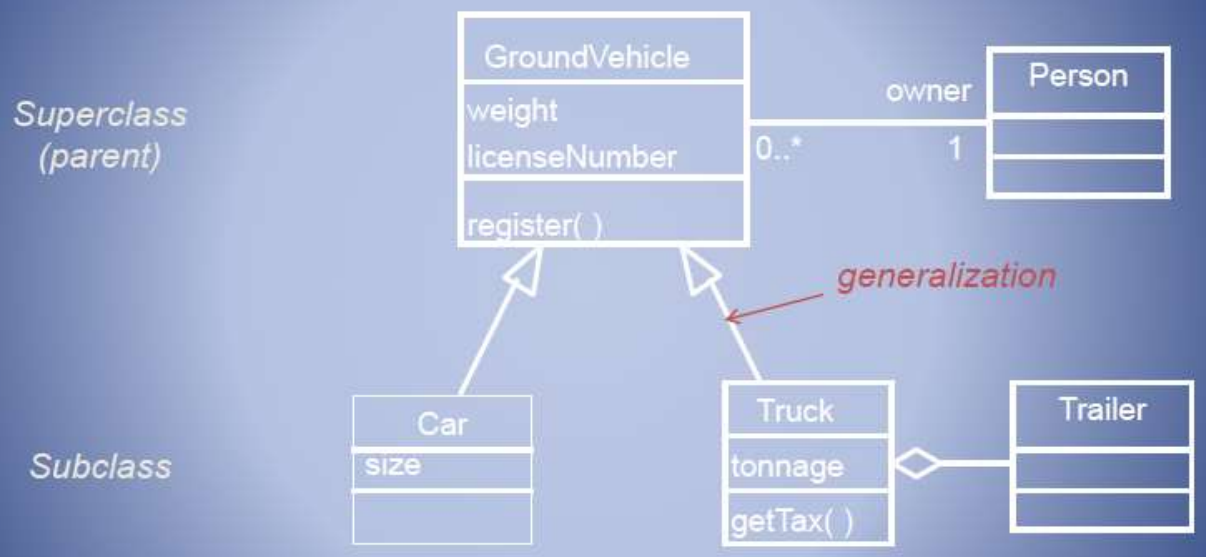
Use multiple inheritance only when needed, and always with caution !

What Gets Inherited?

- A subclass inherits its parent's attributes, operations, and relationships
- A subclass may:
  - Add additional attributes, operations, relationships
  - Redefine inherited operations (use caution!)
- Common attributes, operations, and/or relationships are shown at the highest applicable level in the hierarchy

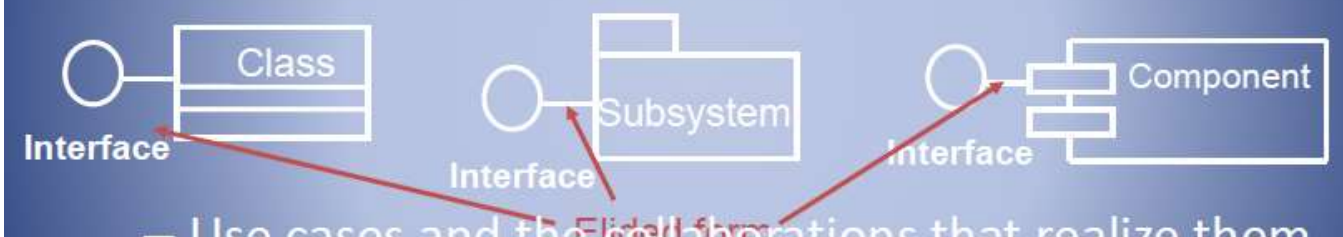
Inheritance leverages the similarities among classes

Example: What Gets Inherited

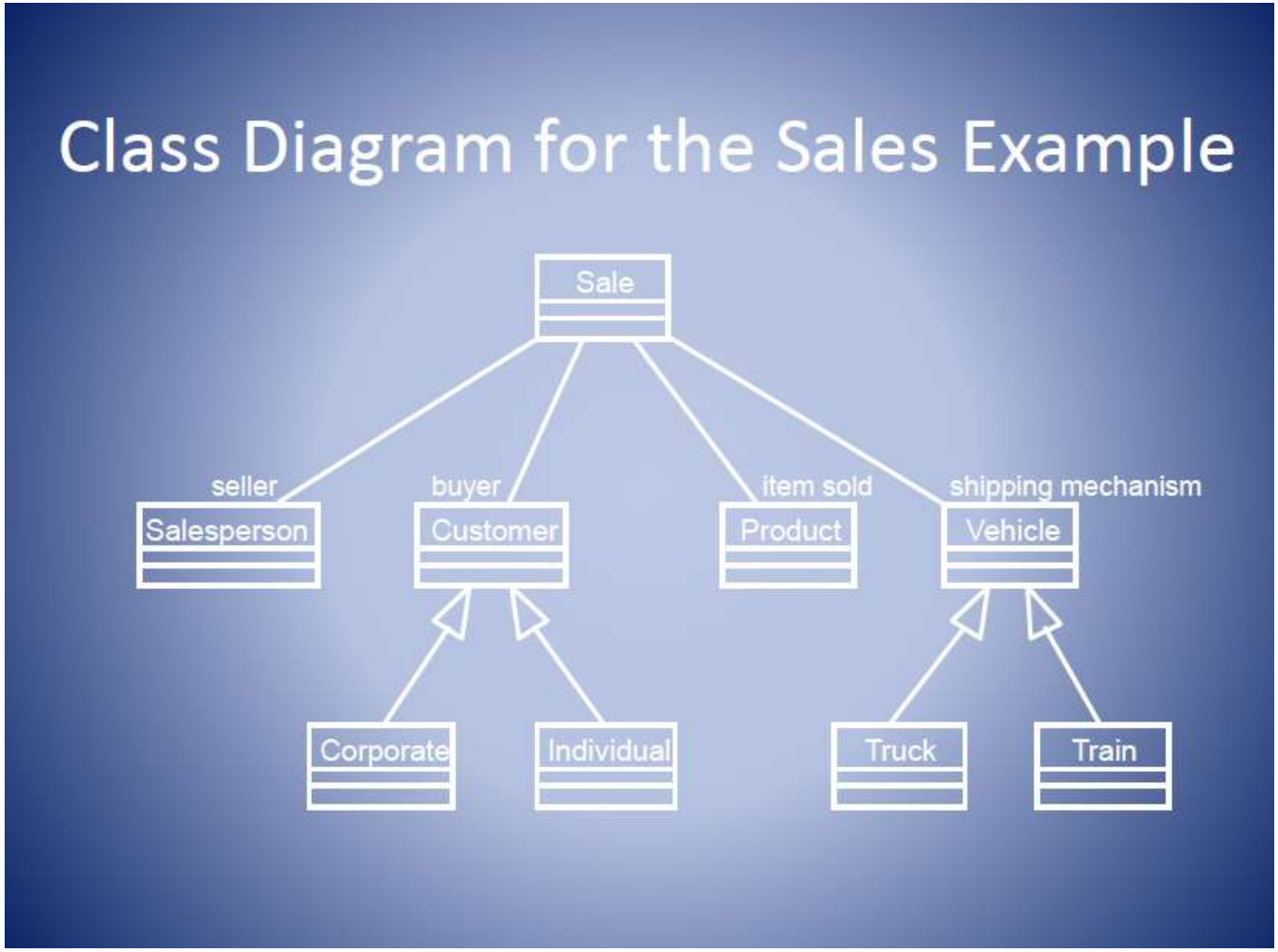


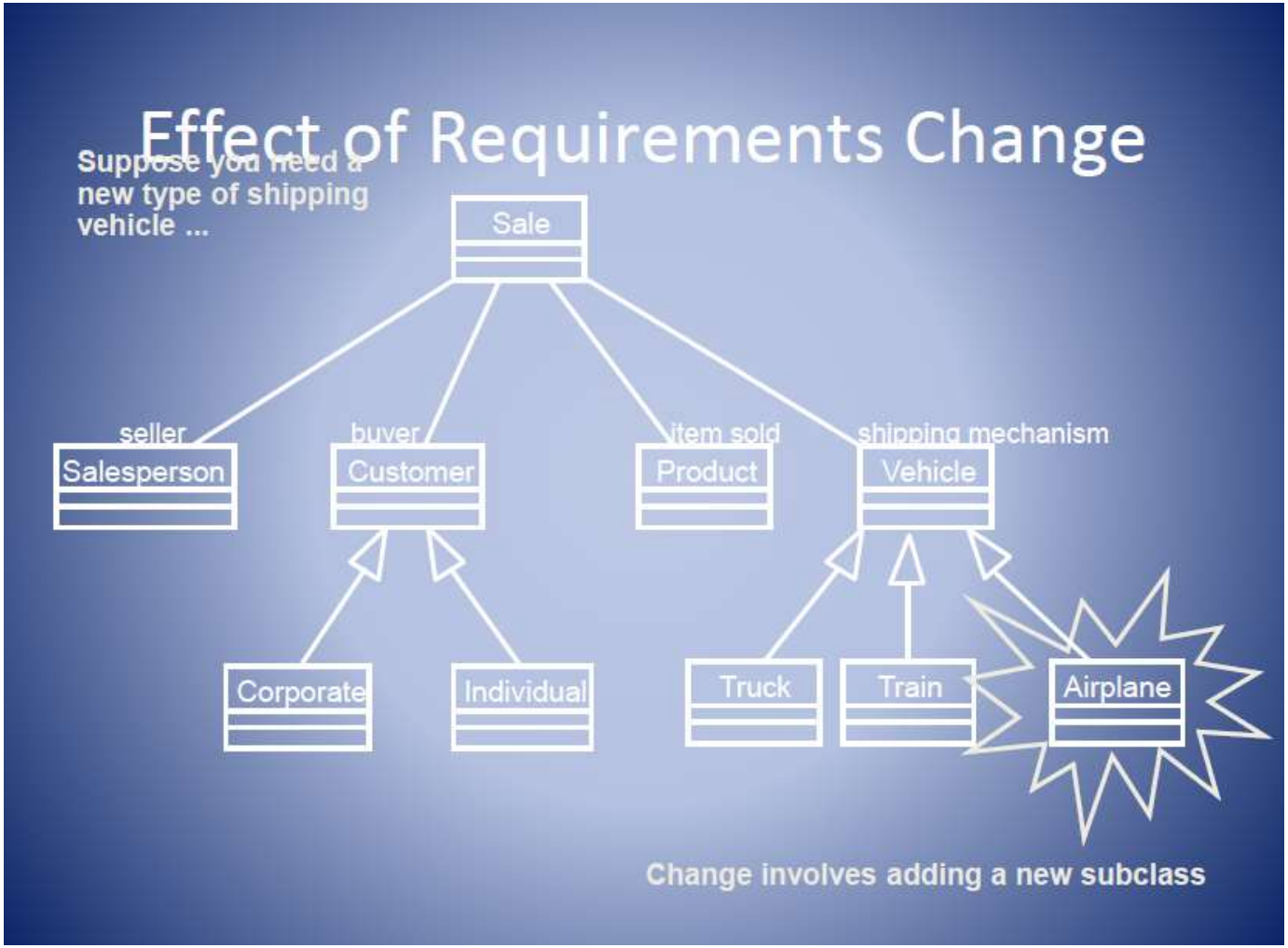
Relationships: Realization

- One classifier serves as the contract that the other classifier agrees to carry out
- Found between:
  - Interfaces and the classifiers that realize them



  - Use cases and the collaborations that realize them


Class Diagram for the Sales Example
Sale
seller
buyer
item sold
shipping mechanism
Salesperson
Customer
Product
Vehicle
Corporate
Individual
Truck
Train


Effect of Requirements Change
Suppose you need a new type of shipping vehicle ...
Sale
seller
Salesperson
buyer
Customer
item sold
Product
shipping mechanism
Vehicle
Corporate
Individual
Truck
Train
Airplane
Change involves adding a new subclass

Notes

- A note can be added to any UML element
- Notes may be added to add more information to the diagram
- It is a 'dog eared' rectangle
- The note may be anchored to an element with a dashed line