UML Class Diagrams

CSCI-3081: Program Design and Development

References for this lecture / examples adapted from:
1. Fowler Text
2. http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/
3. http://atlas.kennesaw.edu/~dbraun/csis4650/A&D/UML_tutorial/class.htm

Touching Base on Your Weekly Writing Assignment.

Today

- UML Class Diagrams
  - How to read them
  - How to create them

UML Basics

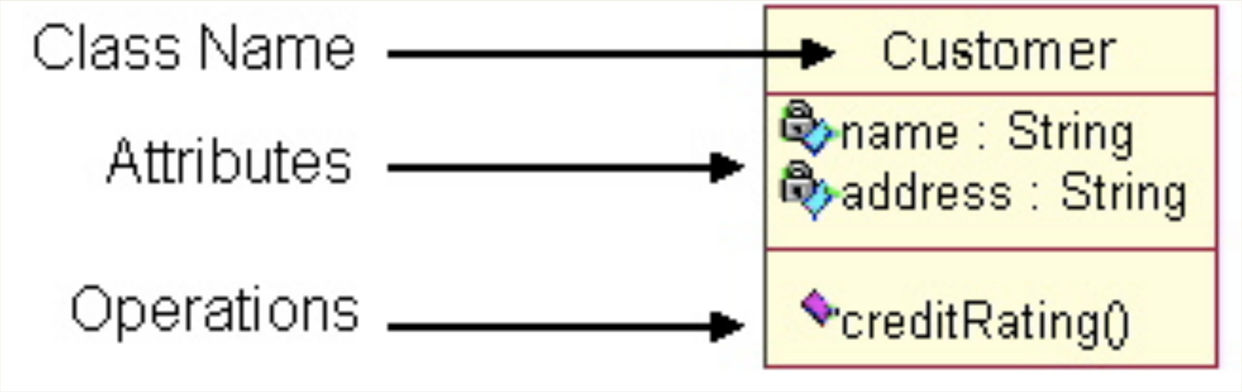
- Unified Modeling Language
- A graphical notation for capturing design information
- Used for designing and for conveying designs to others
- How closely do UML diagrams correspond to a program’s implementation?

Different Notations within the UML

- Notations for:
  - Describing classes and their relationships
  - Describing interactions between classes
- Examples:
  - Class Diagrams
  - Sequence Diagrams
  - State Machine Diagrams
  - Activity Diagrams
- Class Diagrams are the place to start.

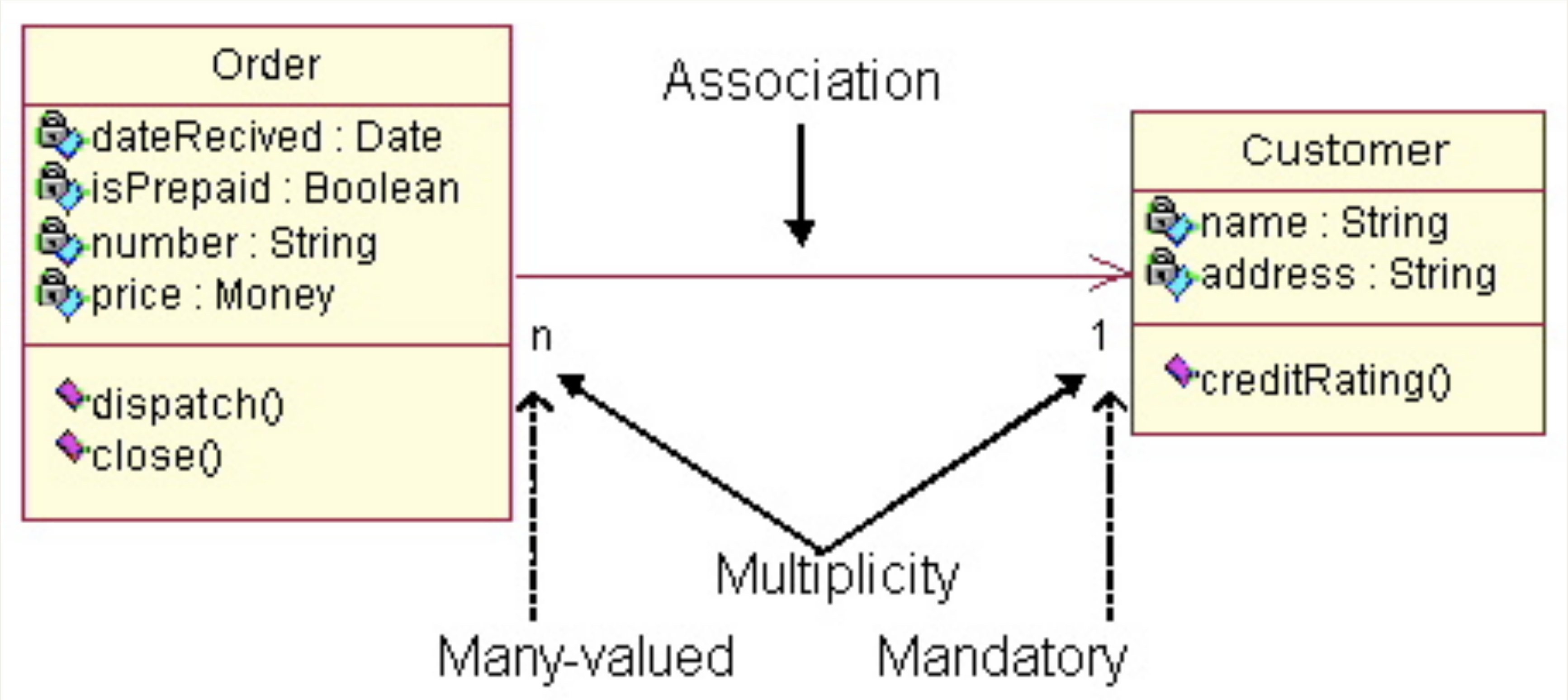
Class Diagrams

- Describe types of objects and static relationships between them.
- Boxes represent classes.
- Classes are composed of three things:
  - a name
  - attributes
  - operations



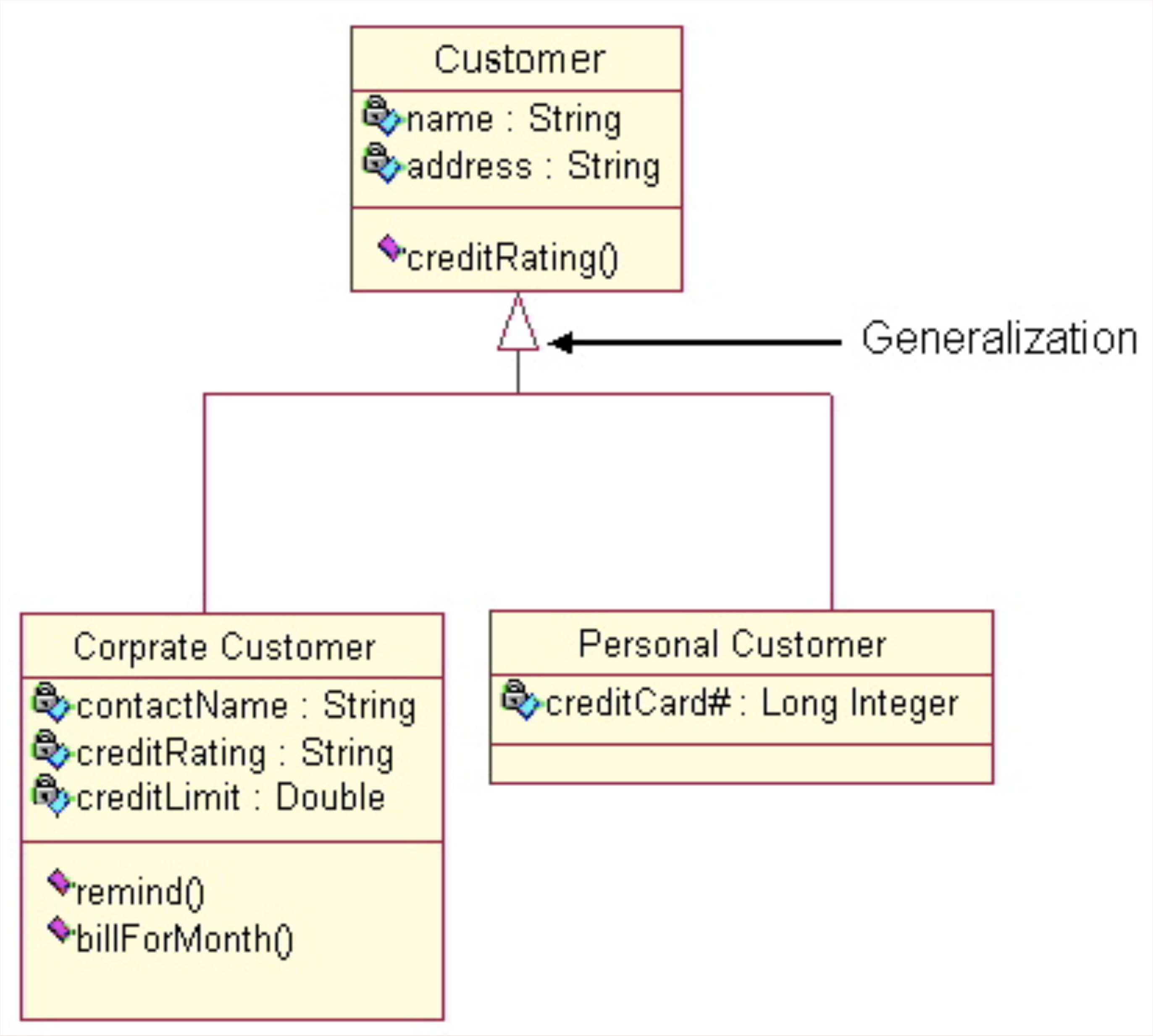
Class Diagrams (2)

- Class Diagrams can also show relationships, such as containment, inheritance, associations, etc.
- Example of association:



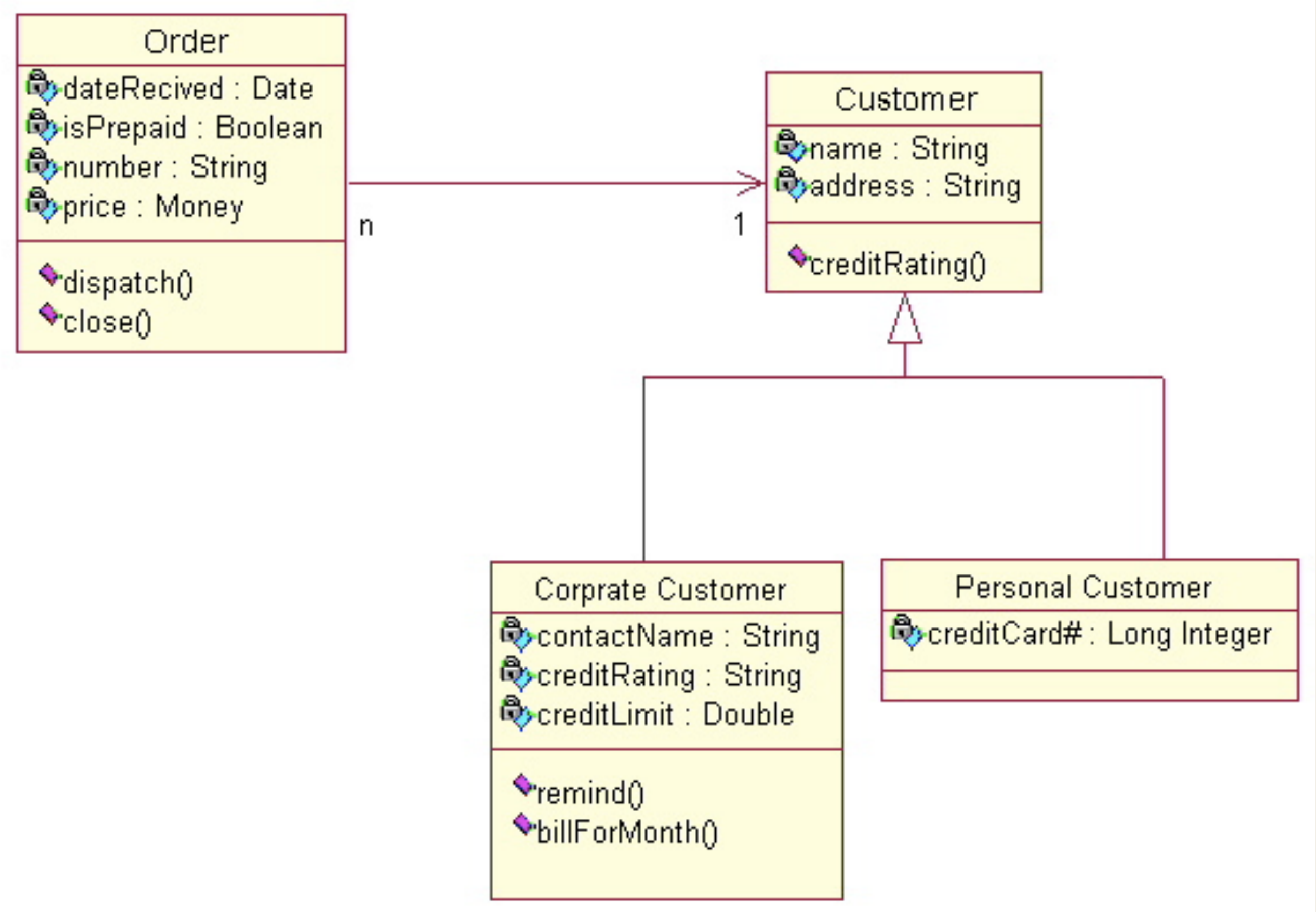
Class Diagrams (3)

- Another important class relationship is generalization, or inheritance.
- Example:



Class Diagrams (4)

- All together:



Order

dateRecived : Date
isPrepaid : Boolean
number : String
price : Money

dispatch()
close()

n

1

Customer

name : String
address : String

creditRating()

Corprate Customer

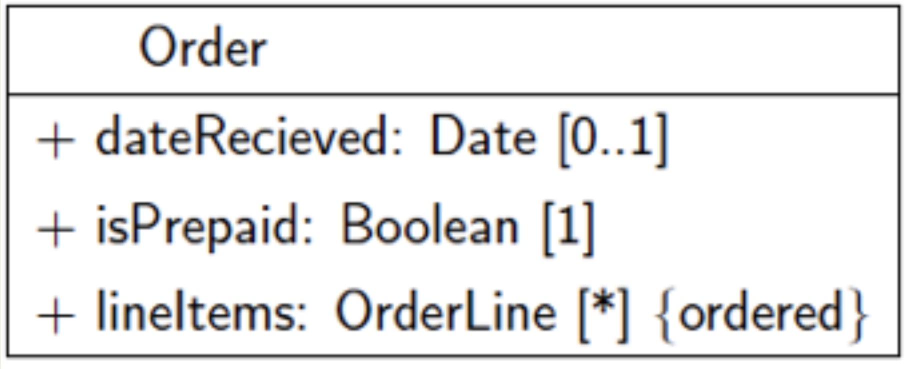
contactName : String
creditRating : String
creditLimit : Double

remind()
billForMonth()

Personal Customer

creditCard# : Long Integer

Specific Formatting of UML Class Diagrams



Order

+ dateRecieved: Date [0..1]

+ isPrepaid: Boolean [1]

+ lineItems: OrderLine [*] {ordered}

- Top section: class name
- Second section: attributes on separate lines
- Third section: left blank in this example -- that's allowed.
- Formatting for the attributes section:

  visibility name : type multiplicity = default {property-string}

  - **visibility**: + for public, - for private
  - **name** and **type** as you would write in implementing the class
  - **multiplicity** indicates the number of instances of objects
  - **= default** specifies a default value
  - **{property-string}** can be filled with a number of properties, e.g. readOnly.

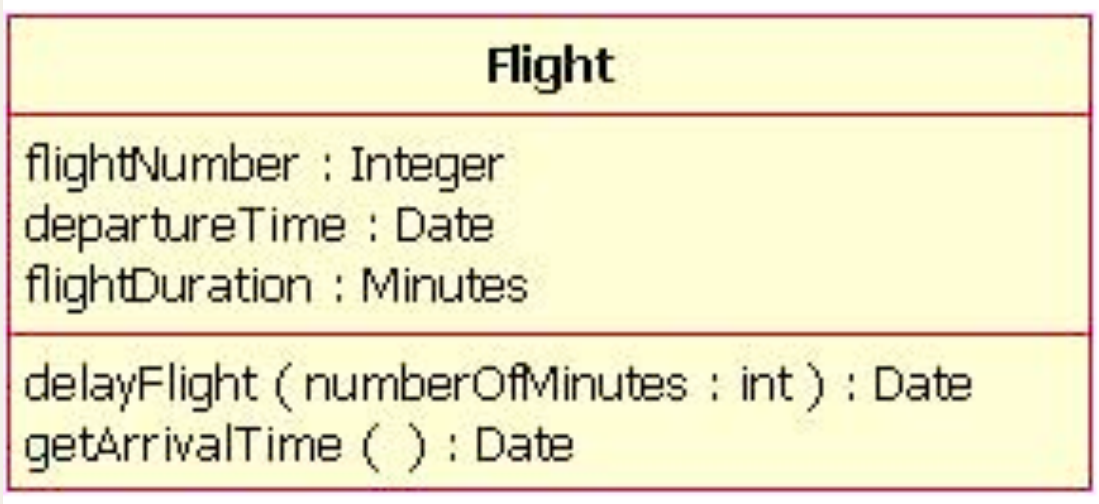
Multiplicity Indicators

Indicator	Multiplicity
0..1	zero or one
1	one only (default)
0..*	zero or more
1..*	one or more
n	only n (where n > 1)
0..n	zero to n (where n > 1)
1..n	one to n (where n > 1)
*	zero or more

Multiplicity: How would you write these?

- Optional single-valued
- Optional multi-valued
- Mandatory single-valued
- Mandatory multi-valued

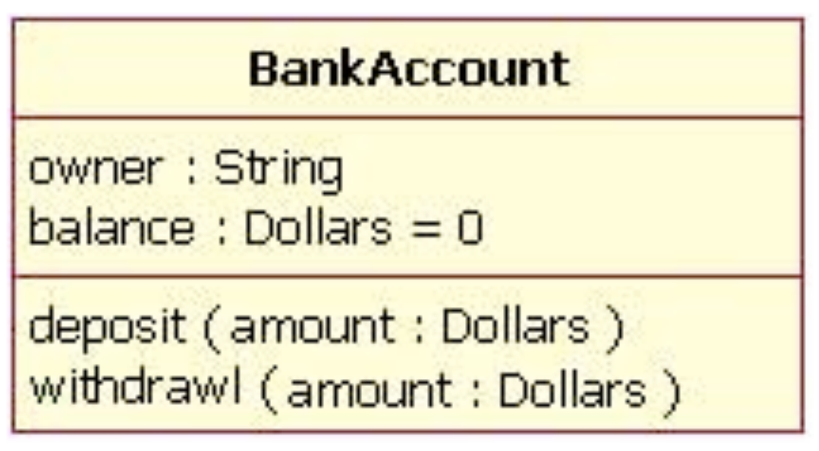
Class Operations List



Flight

flightNumber : Integer
departureTime : Date
flightDuration : Minutes

delayFlight (numberOfMinutes : int) : Date
getArrivalTime ( ) : Date



BankAccount

owner : String
balance : Dollars = 0

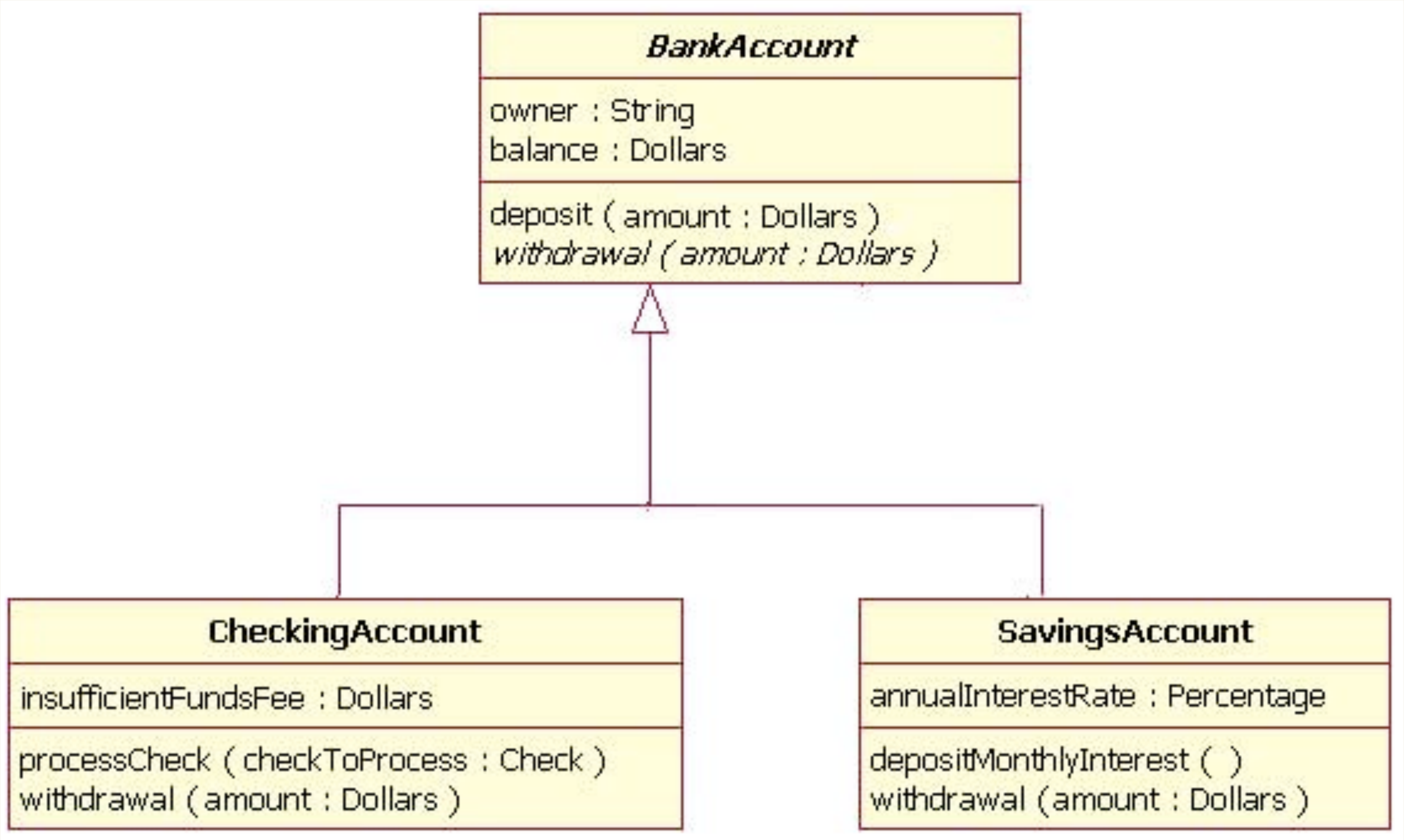
deposit (amount : Dollars)
withdrawl (amount : Dollars)

- Third section is for class operations (methods).
- Notation:

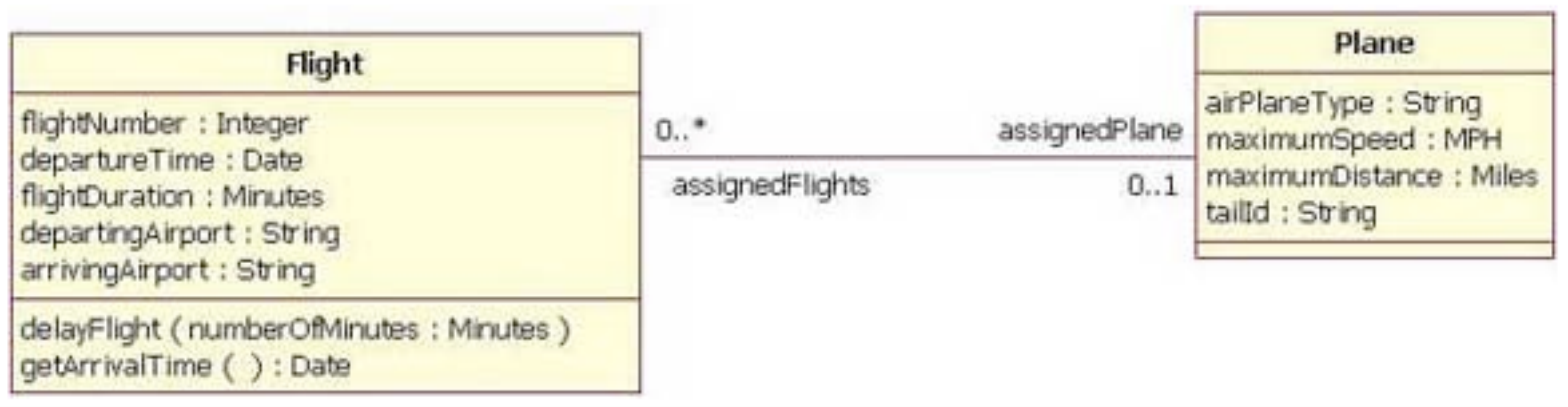
name(parameter list) : type of value returned

Inheritance

- A bank account example, different withdrawal methods for checking and savings:

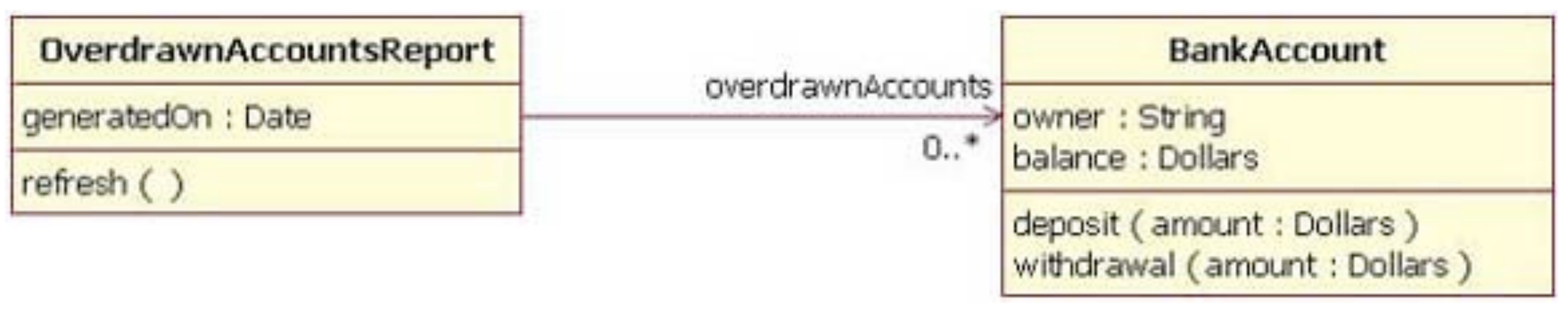

BankAccount
owner : String
balance : Dollars
deposit (amount : Dollars)
withdrawal (amount : Dollars)
CheckingAccount
insufficientFundsFee : Dollars
processCheck (checkToProcess : Check)
withdrawal (amount : Dollars)
SavingsAccount
annualInterestRate : Percentage
depositMonthlyInterest ( )
withdrawal (amount : Dollars)

Associations in More Detail



- Associations are linkages between two classes.
- (For smaller, less significant classes, you could skip drawing an association and instead just list the this as an attribute in the 2nd box.)
- By default, associations are bi-directional, meaning both classes know about the association.
- Indicated by a solid line between the two classes. Put a role and a multiplicity at each end of the line.

Unidirectional Association



- Indicated by a solid line with an open arrowhead pointing to the known class.
- Here the BankAccount class has no idea that it is associated with an OverdrawnAccountsReport, but the report knows about some number (0 to *) of bank accounts.

Discussion

- Just how Unified is the Unified Markup Language?

A Current, Real Example from My Work

Other Important Class Diagram Ideas

- Abstract methods and class names should be italicized. (See *BankAccount* example on previous slides.)
- Static features should be underlined:

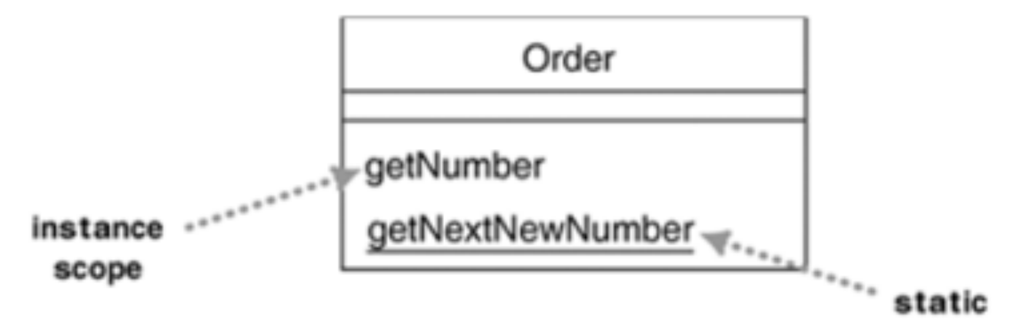


Figure 5.2. Static notation

- You can include comments, for example to indicate the Responsibility of each class:

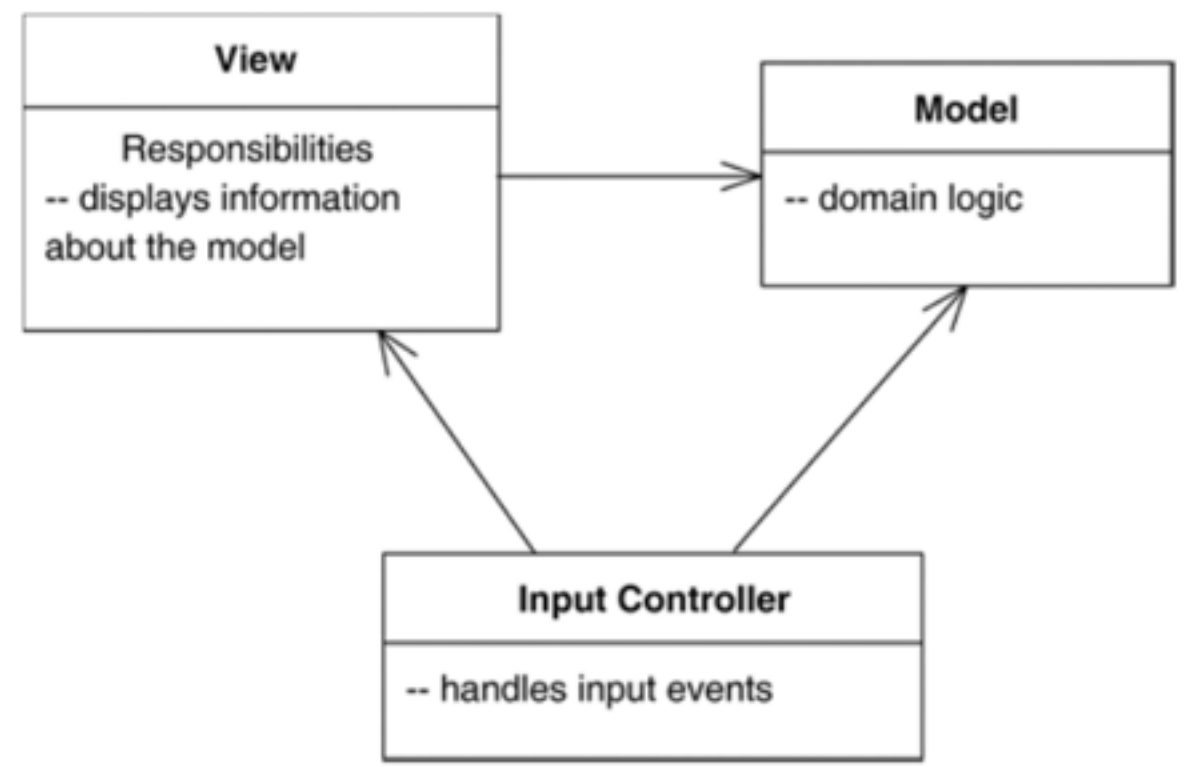
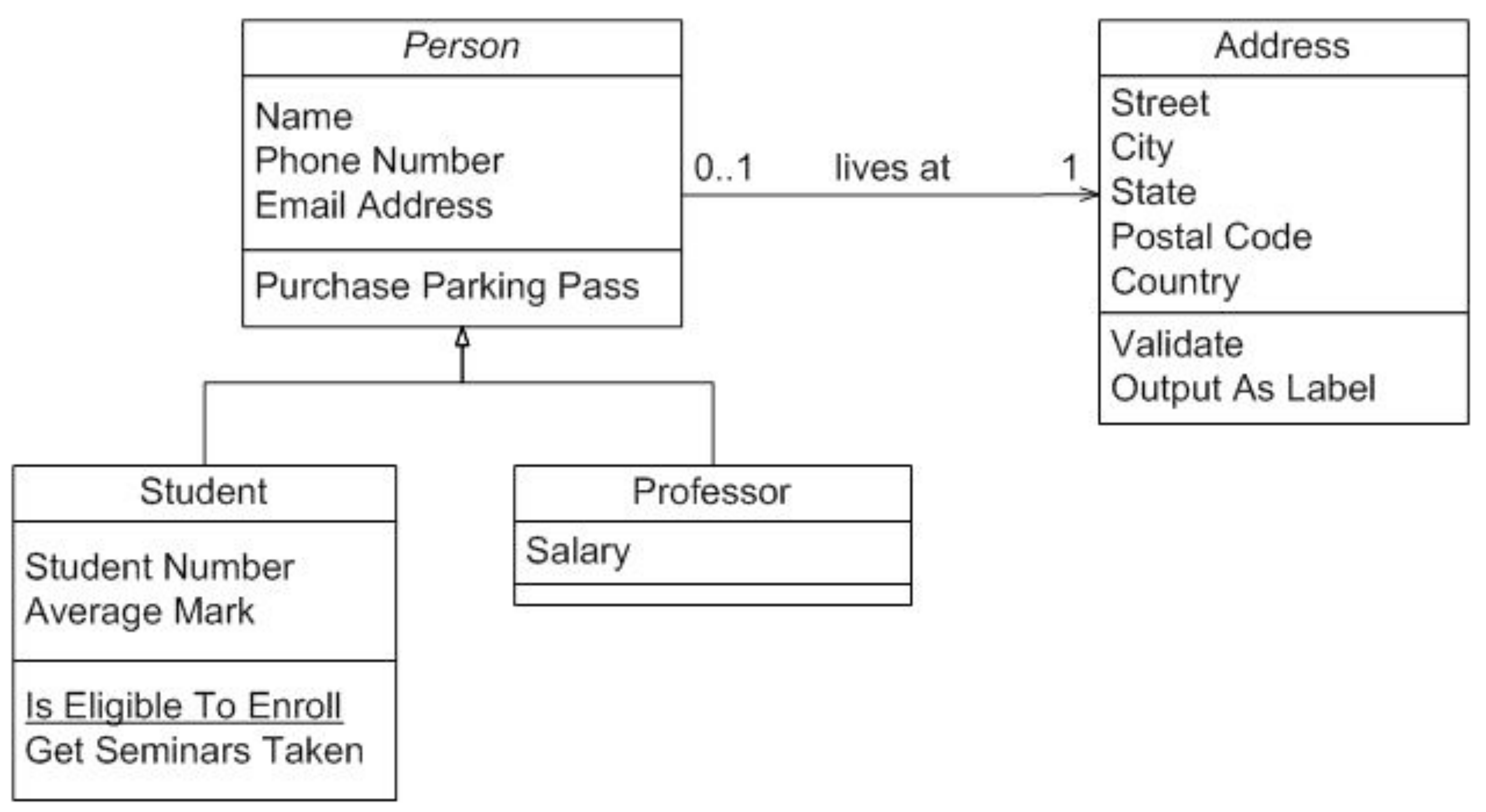
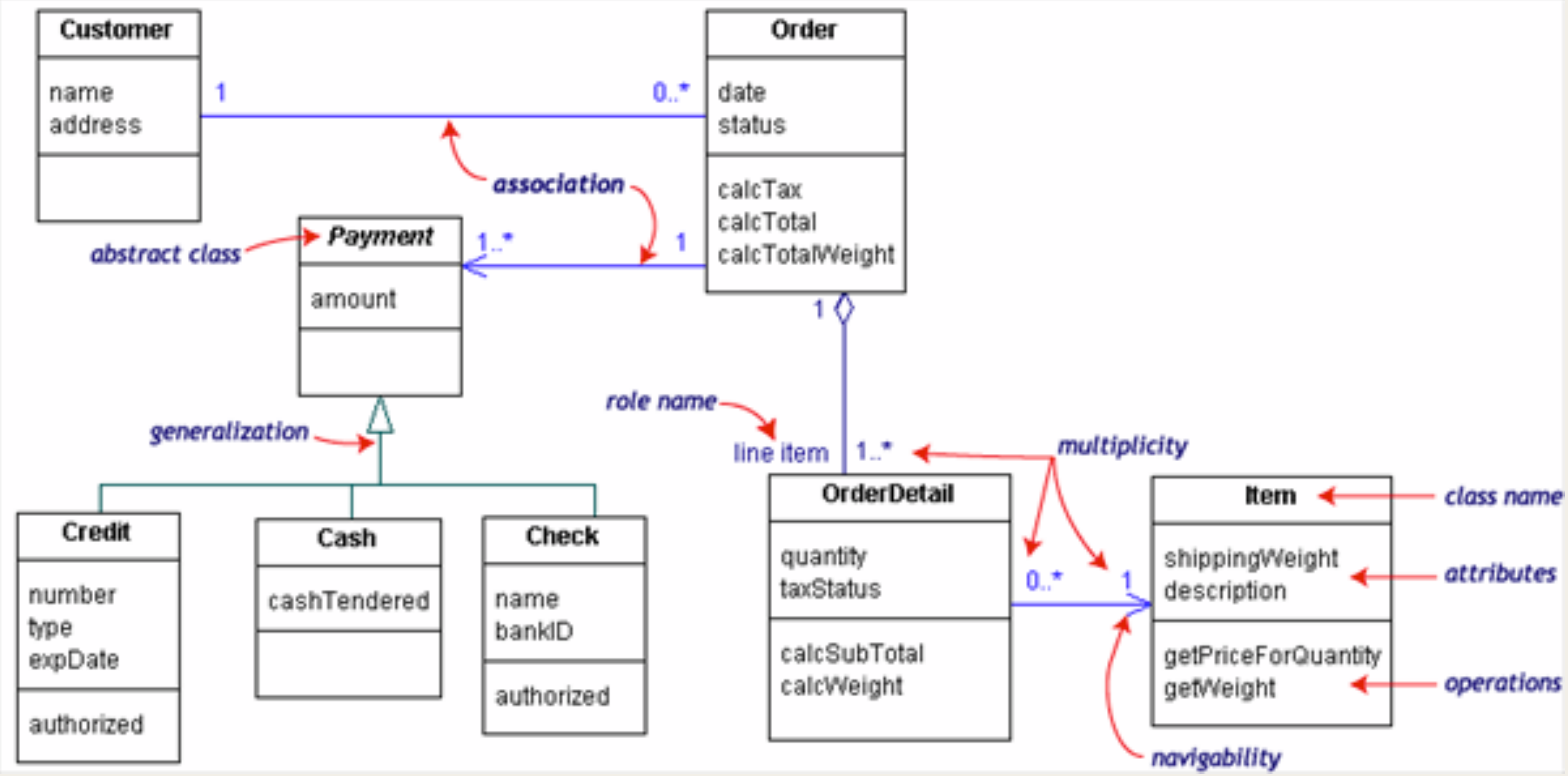
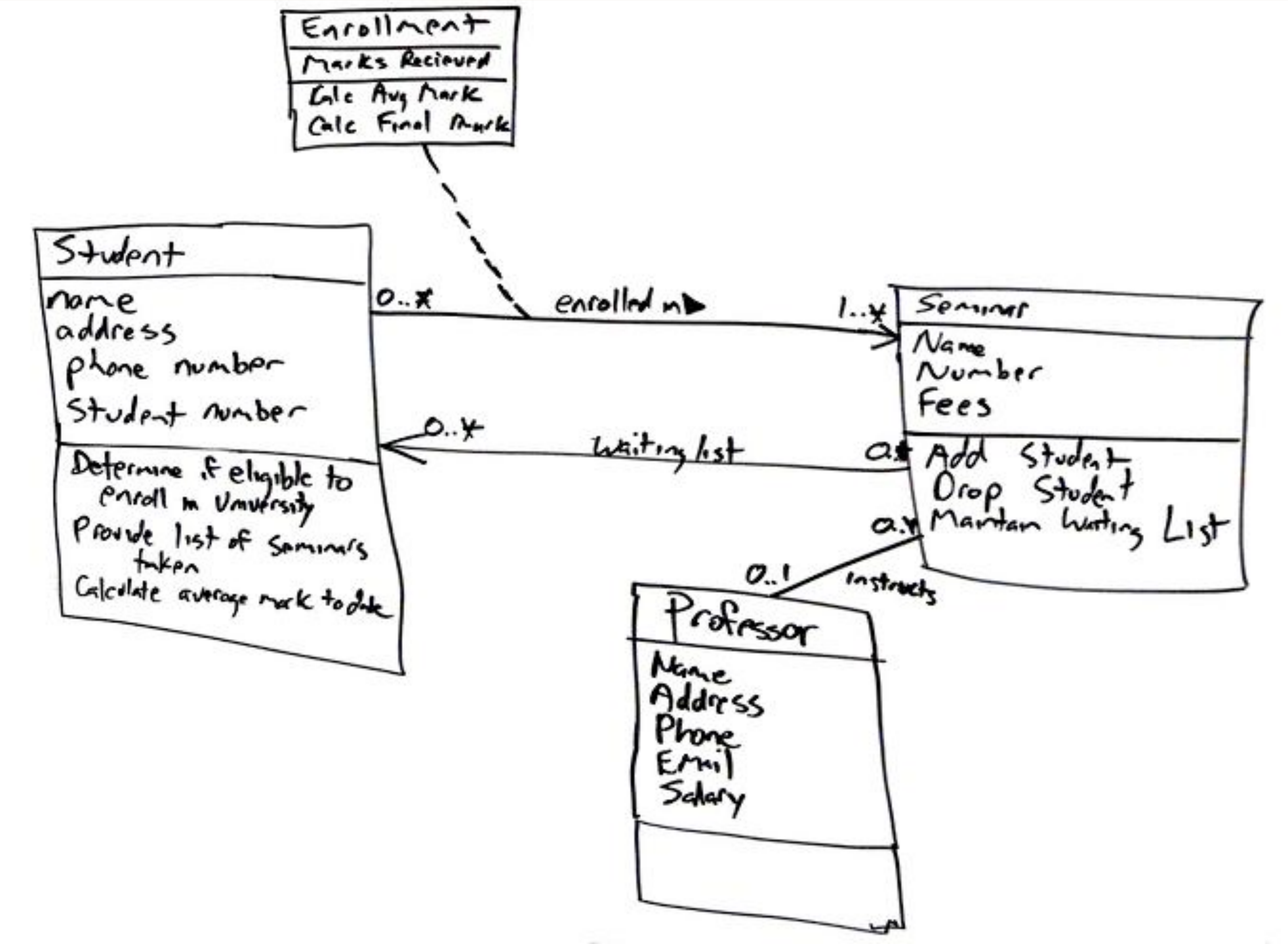


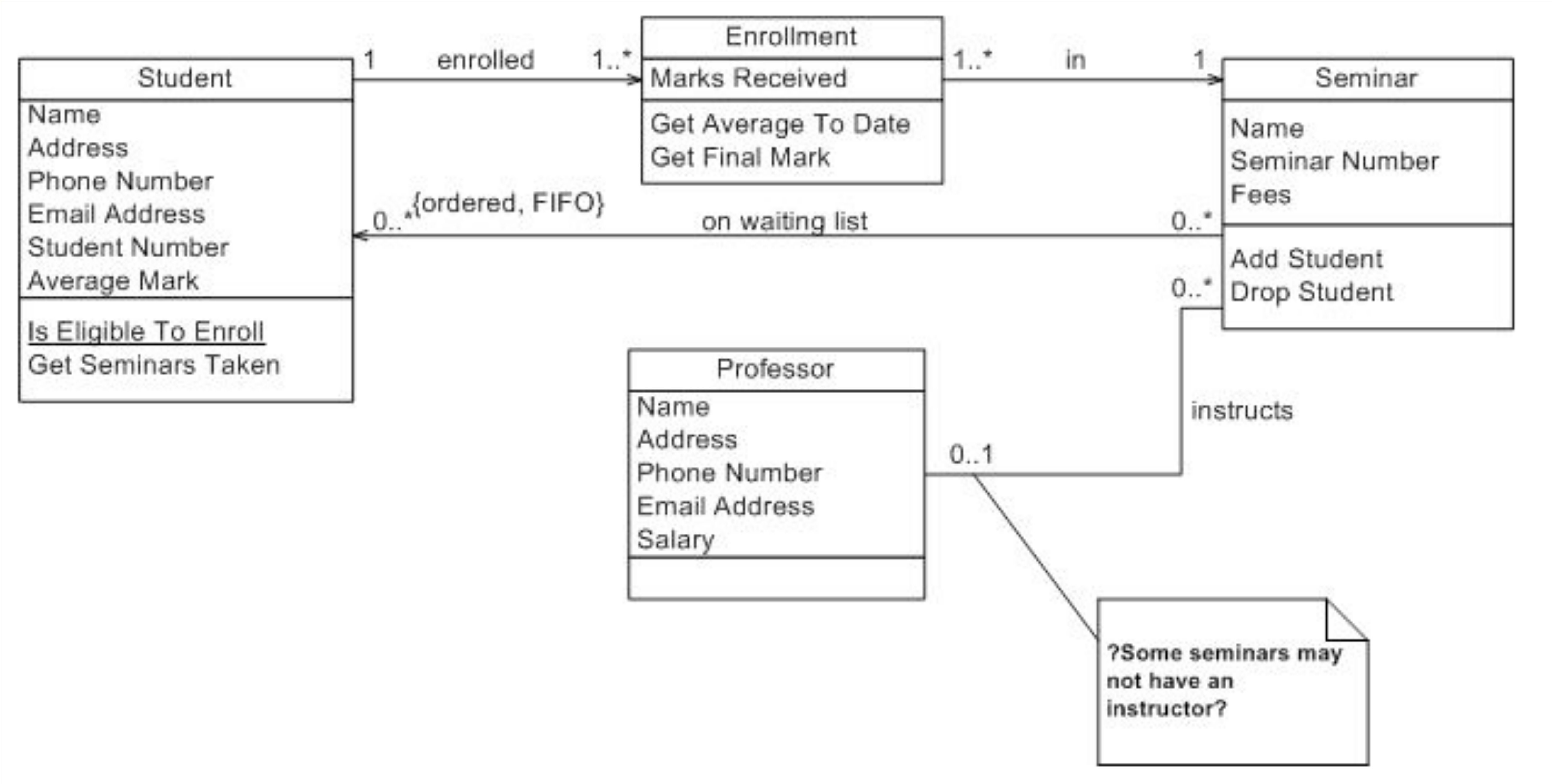
Figure 5.1. Showing responsibilities in a class diagram

Examples


Person
Name
Phone Number
Email Address
Purchase Parking Pass
0..1
lives at
1
Address
Street
City
State
Postal Code
Country
Validate
Output As Label
Student
Student Number
Average Mark
Is Eligible To Enroll
Get Seminars Taken
Professor
Salary


Customer
name
address
1
0..*
Order
date
status
calcTax
calcTotal
calcTotalWeight
association
abstract class
Payment
1..*
1
amount
1
role name
generalization
line item
1..*
multiplicity
Credit
number
type
expDate
authorized
Cash
cashTendered
Check
name
bankID
authorized
OrderDetail
quantity
taxStatus
calcSubTotal
calcWeight
0..*
1
Item
class name
shippingWeight
description
attributes
getPriceForQuantity
getWeight
operations
navigability


Enrollment
Marks Recieved
Calc Avg Mark
Calc Final Mark
Student
name
address
phone number
Student number
Determine if eligible to enroll in University
Provide list of Seminars taken
Calculate average mark to date
0..*
enrolled in ▶
1..*
Seminar
Name
Number
Fees
Add Student
Drop Student
Maintain Waiting List
0..*
waiting list
0..*
0..*
0..1
instructs
Professor
Name
Address
Phone
Email
Salary



?Some seminars may not have an instructor?

Tools

- http://martinfowler.com/bliki/UmlSketchingTools.html
- http://www.gliffy.com/uses/uml-software/
- http://alexdp.free.fr/violetumleditor/page.php
- http://www.umlet.com/umlet_13_1/umlet_13.1.zip
- … (many more on the web)

Please check Moodle frequently to stay on track:

- Read Fowler Chp 3 (if not already).
- First lab tomorrow.
- Weekly Writing Assignment due Mon at midnight.
- Office hour times announced on Moodle.