31269: Business Requirements Modeling

Week 8 Lecture:

Object Oriented Models - Class Modelling

Object Oriented Modelling

- Last Half: Structured Analysis
  - What info do we need; where to get it?
  - How do we get the information we need?
  - How do we analyse the information?
- This Half: Object Oriented Analysis
  - Use Case Modelling
  - Class Modelling (***This Lecture***)
  - Interaction Modelling
  - State and Event Modelling

Objectives

- Appreciate how Object Oriented (OO) modelling techniques can help to understand the working of business systems
- Discover why system specifications are important and how OO modeling can be used to specify systems and user requirements
- Use object oriented system analysis techniques to develop a system model (**Class Diagram/Model**)

Topics

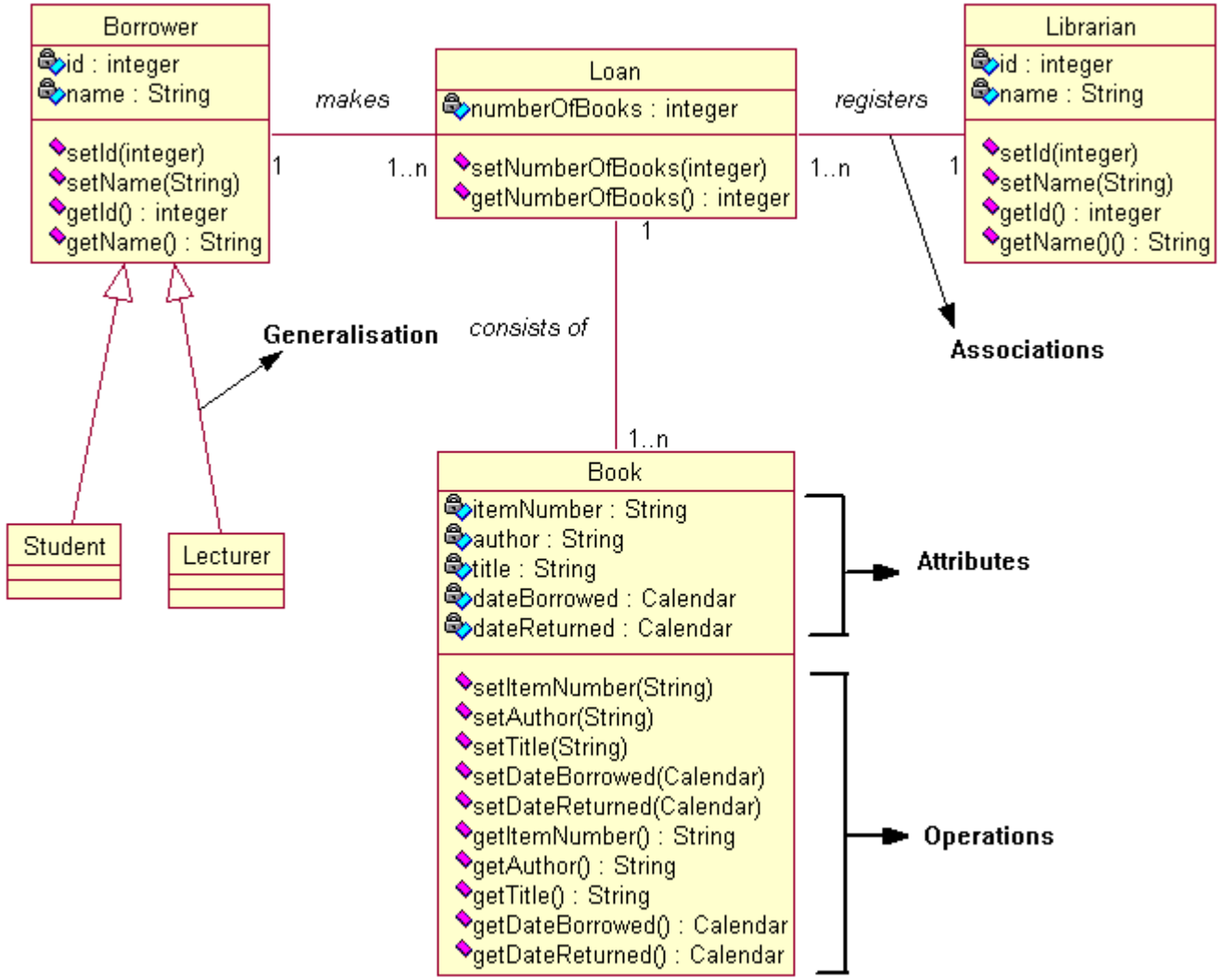
- Components of a Class Diagram
- Relationship between Classes
- Rules for Class Diagram

Class Diagram

- A **Class Diagram** is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among classes/objects.
- The class diagram is the main building block of object oriented modelling.
- It is used both for general conceptual modelling of the application, and for detailed modelling translating the models into programming code.
- Class diagram is not only used for **visualizing, describing and documenting different aspects of a system** but also for constructing executable code of the software application.

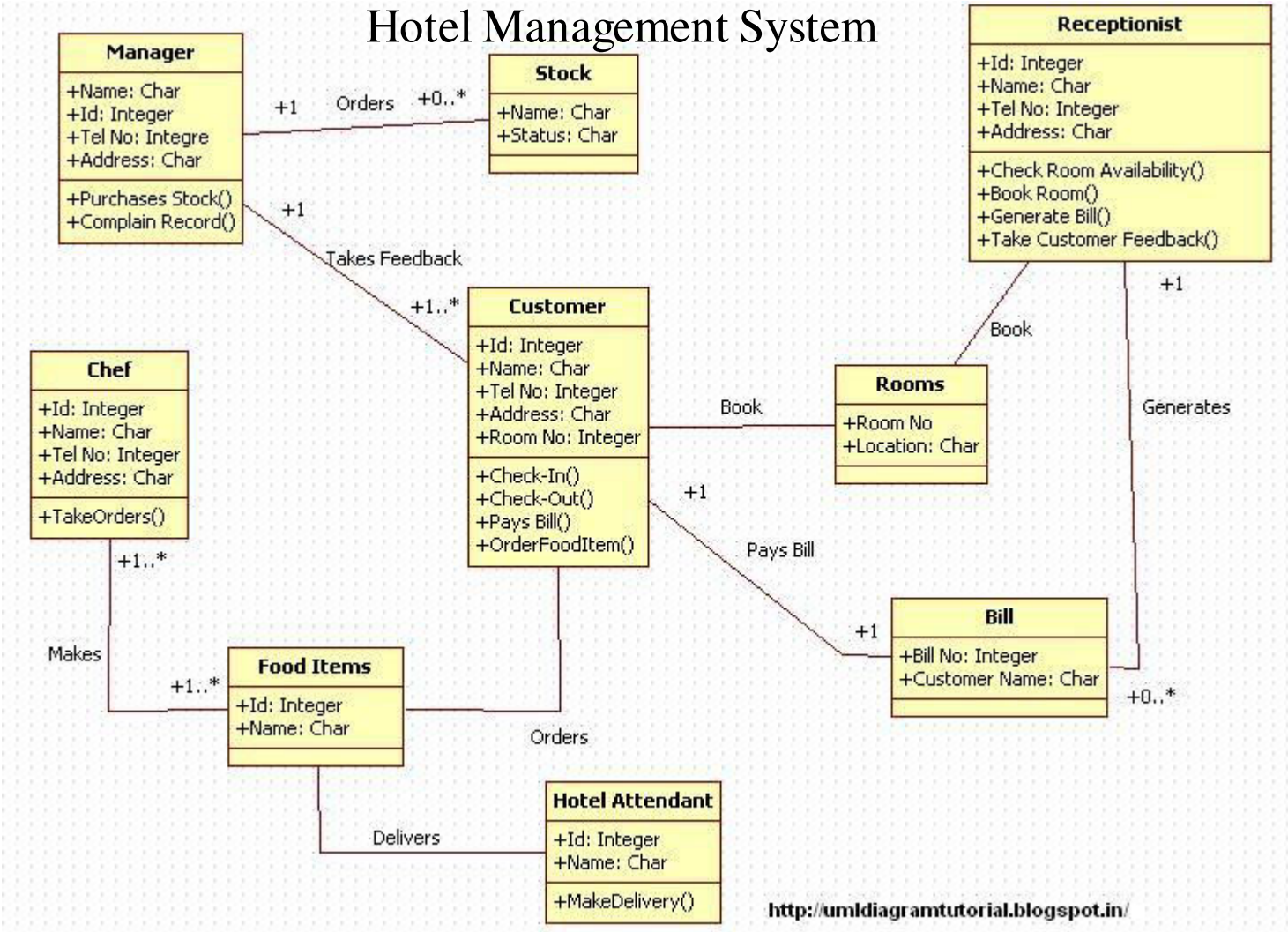
Source: https://en.wikipedia.org/wiki/Class_diagram

Class Diagram – Example 1

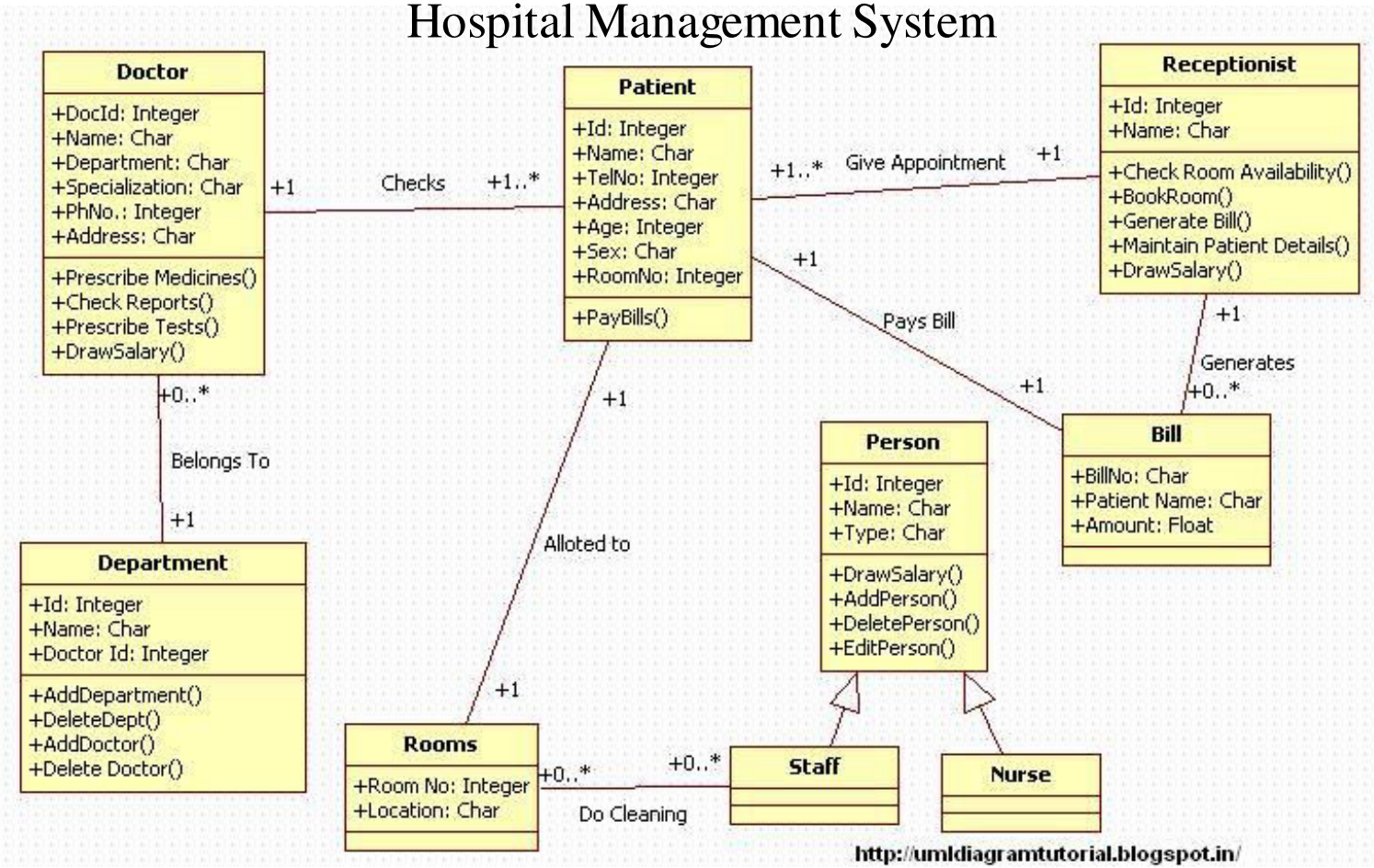


Source: http://www.csse.monash.edu.au/courseware/cse3302/2001/lectures/lecture03/notes03a.htm

Class Diagram Example 2



Class Diagram Example 3

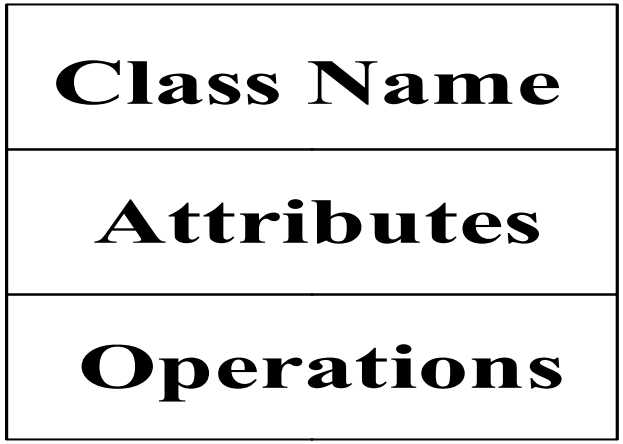


Class

- A class is an **abstract definition** of an object.
- A class is a **plan** or a **blueprint** from which one or more objects can be created. Whereas an object represents real world things, both tangible and intangible.
- A class is a description of a set of objects that share the same:
  - Attributes
  - Operations
  - Relationships
- Graphically, a class is rendered as a rectangle, usually including its name, attributes, and operations in separate, designated compartments.

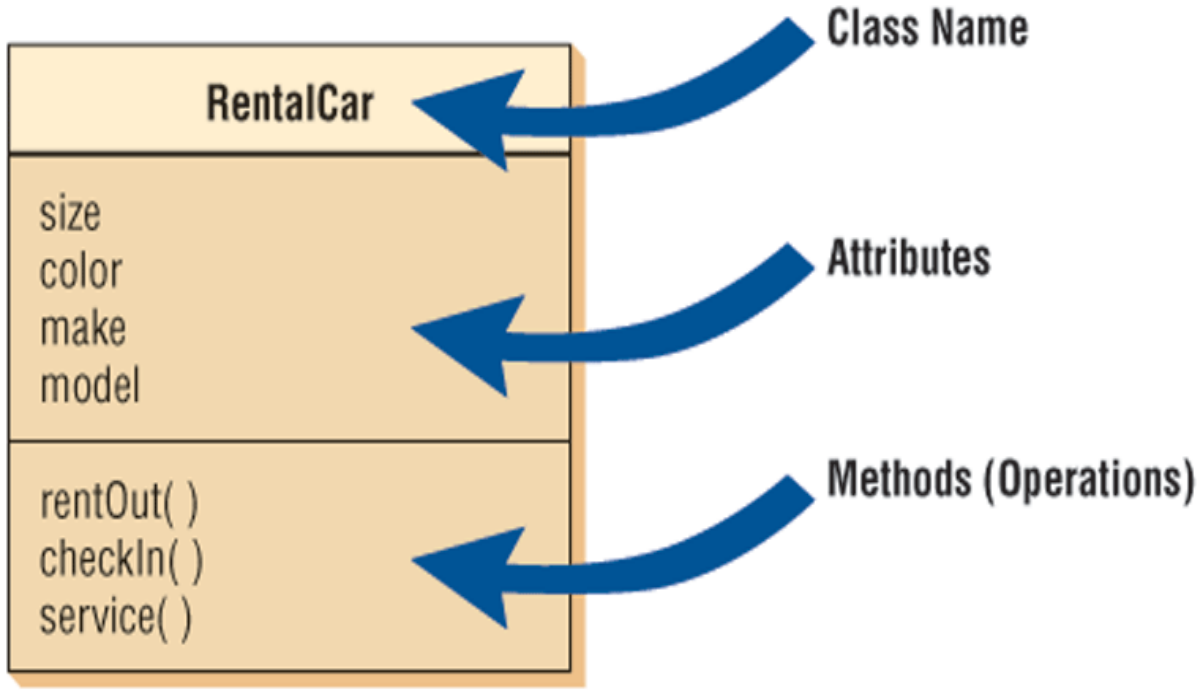
Class

- A class is comprised of attributes and operations (methods) that manipulate these attributes.
- Real-world objects share two characteristics: They all have state (attributes/fields) and behavior (methods/operations).



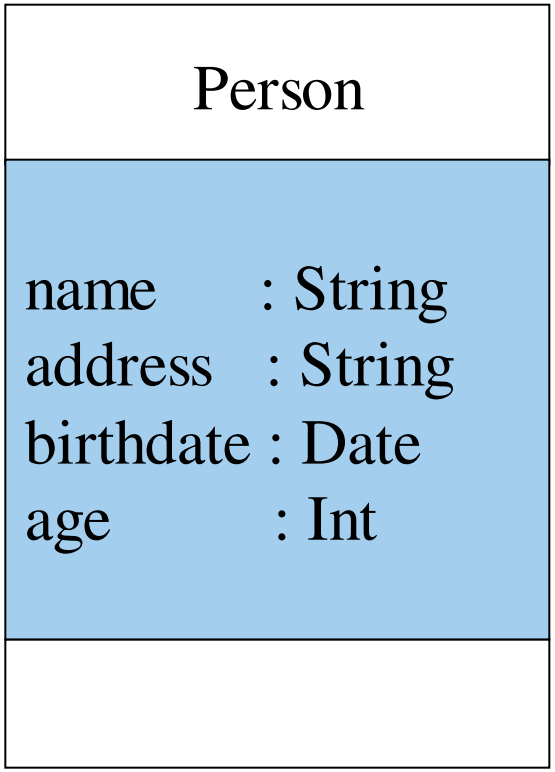
Classes

Figure 18.1 An example of a UML class. A class is depicted as a rectangle consisting of the class name, attributes, and methods.



Attributes/fields

- Attributes: Attributes describe the **characteristics** or are a **description** of a class. In the class diagram, attributes appear in the second compartment just below the name-compartment.

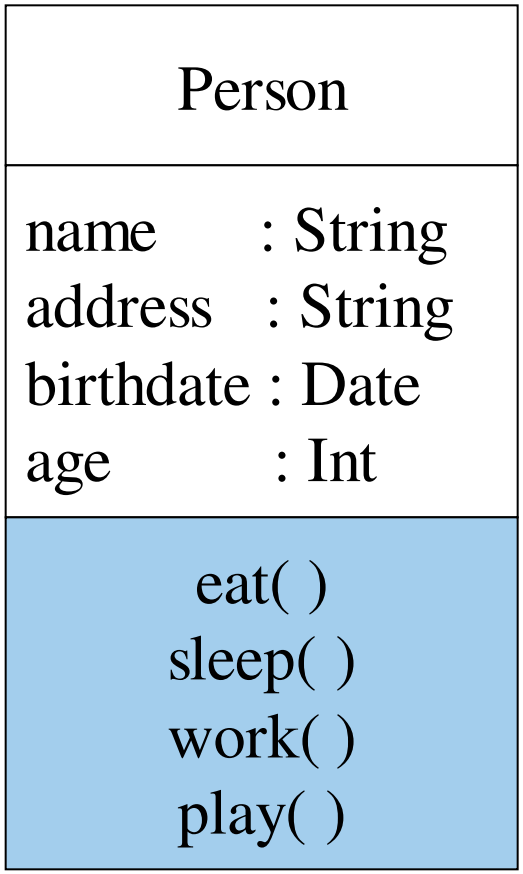


- A class stores its **state** in attributes and the keyword to identify attributes from a specification is **'has'**, indicating what a class has.

- You can specify the data type for each attribute.

Class Operations/Methods

Operations describe the **behavior** of a class and they appear in the third compartment.



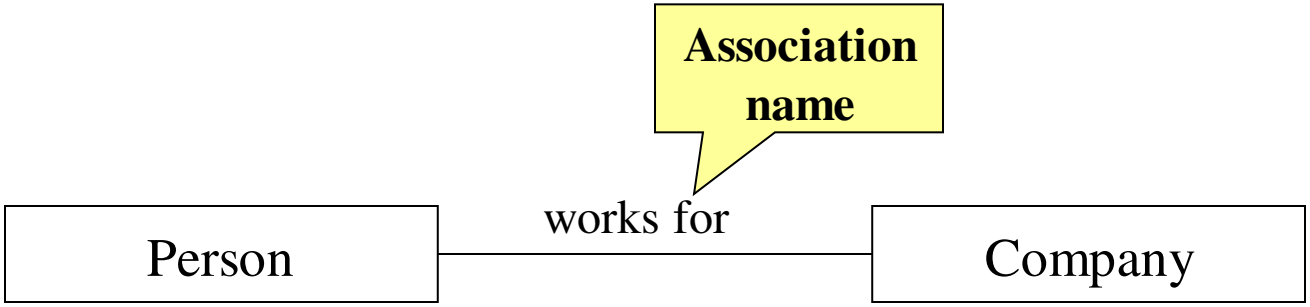
- They are **actions** that can be carried out by, or on, a class/object and the keyword to identify methods from a specification is **'can'**, indicating what an object can do.
- You can specify an operation by stating its signature: listing the name, type, and default value of all parameters.

Relationships

- There are four kinds of relationships between classes:
  - Associations
  - Aggregation
  - Composition
  - Generalisations

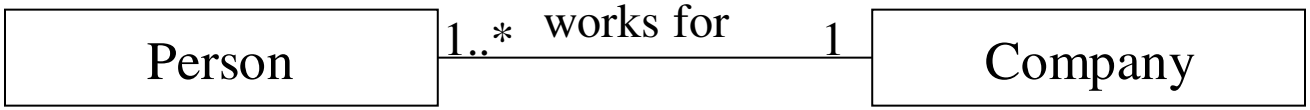
Associations

- An association is a relationship between classes that indicates meaningful and interesting connection.
- An association is represented as a bold unbroken line between two classes.
- Example: “A Person works for a Company”
- To clarify its meaning, an association may be named.
  - The name is represented as a label placed midway along the association line.
  - Usually a verb or a verb phrase.



Associations Multiplicity

- Multiplicity
  - An indication of how many objects may participate in the given relationship.
  - Multiplicity is shown at the ends of an association.



  - A person works for one company.
  - A company has one or more people working in it.

Common Multiplicity

- Multiplicity Indicators

Multiplicity Indicators

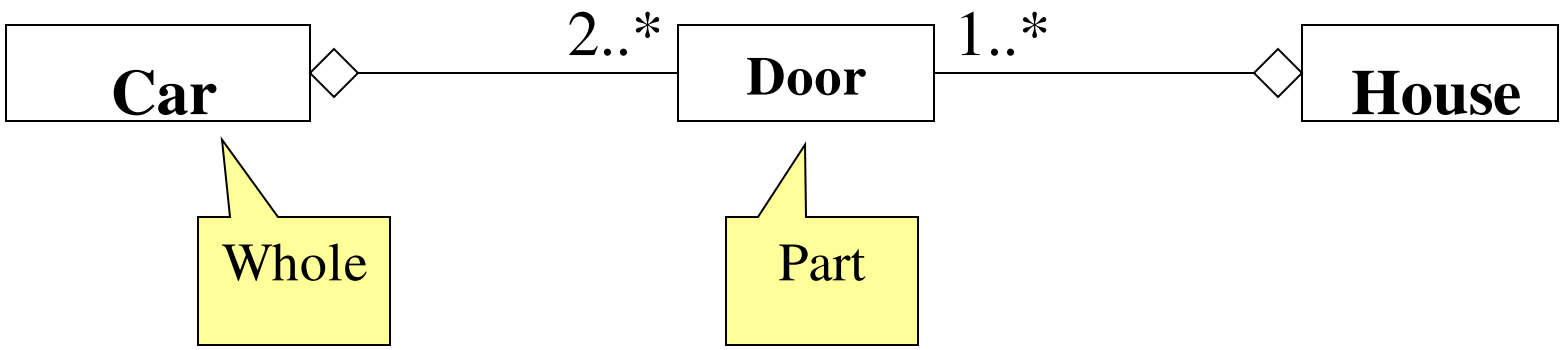
Exactly one	1
Zero or more	(0..*)
One or more	1..*
Zero or one	0..1
Specified range	2..4
Exactly three, five or eight	3,5,8

Aggregation and Composition Relationships

- Aggregation and Composition relationship between classes is a special form of association that models a **whole-part relationship** between the whole and its parts.

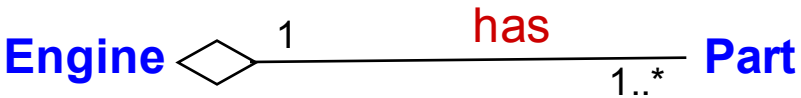
Aggregation

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



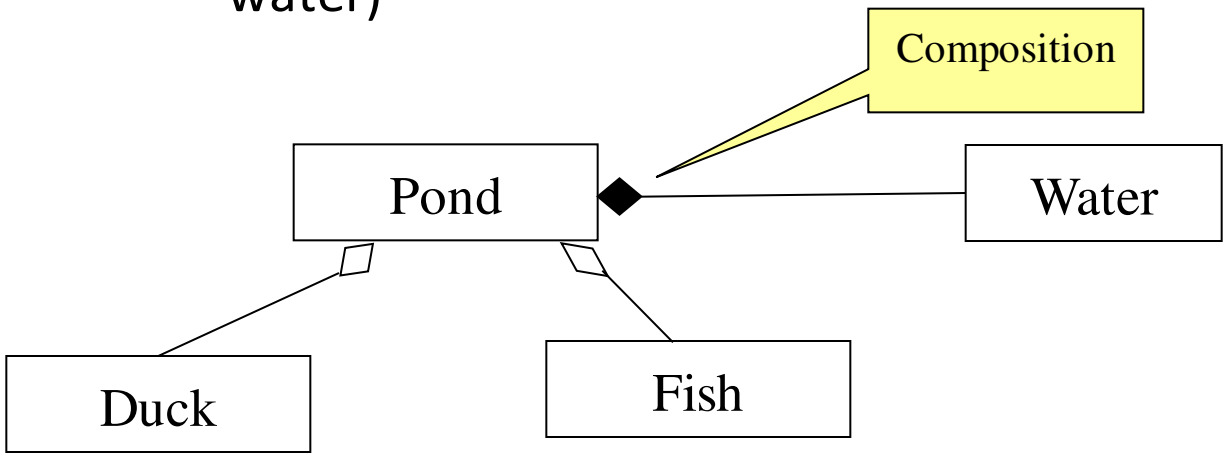
Aggregation

- Aggregation is a “has a” temporary structural relationship (**weak association**).
- Consists of a whole and its parts
- Examples:
  - Library *has* Books
  - Subject *has* Students
  - Constituency *has* Voters
- If the **whole is removed, the part may still exist**.
  - e.g. If Library is closed/removed, the Books may still exist
  - e.g. If Subject is closed/removed, the Students still exist
- Represented by a hollow **white diamond** at the end of the line that is not filled in, points to the class that holds the other class.



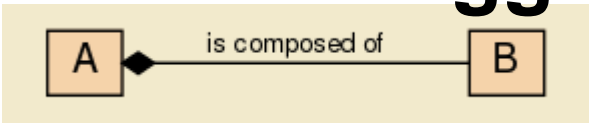
Composition

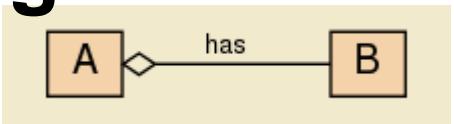
- A **strong form of relationship**.
- Shown with a filled-in **black diamond at the end of a** line.
- The whole has a responsibility for the parts.
- If the **whole is removed, the parts are also removed.**
- The life time of the part is dependent upon the whole.
  - The composite must manage the creation and destruction of its parts. Water is a part-of a Pond. (Pond is a composition of water)



Composition vs Aggregation

- (A and B are classes)

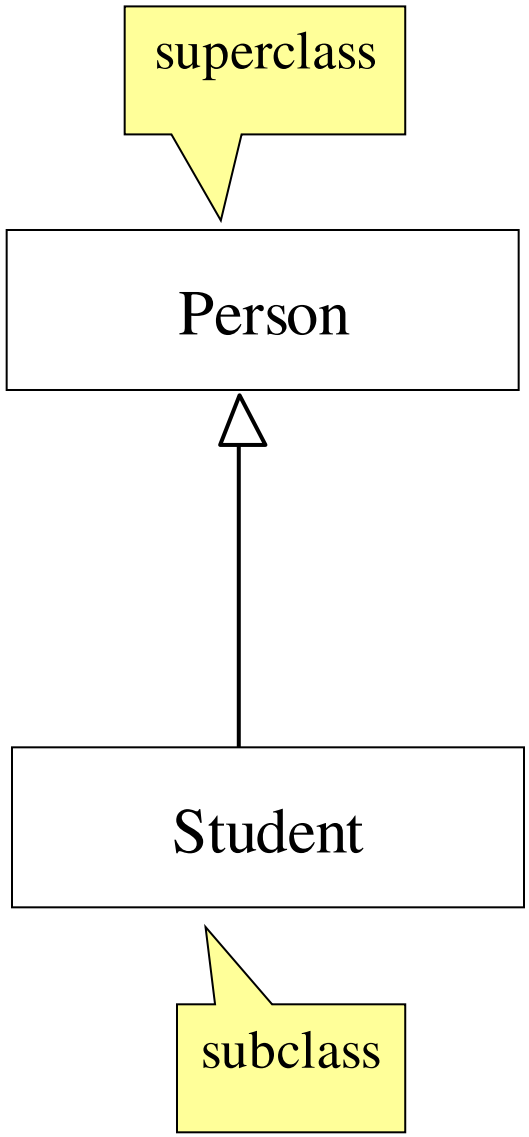




- There is a whole part relationship between classes/objects in both.
- The distinction between aggregation and composition relationship **depends on context of the problem.**
- For example: **Composition:** Water is a part-of a Pond. (Pond is a composition of water) **Aggregation:** Pond has ducks and fishes (Pond aggregates ducks and fishes)
- Example2:
  Composition (Person, Heart, Hand), "**sub objects**" (Heart, Hand) **will be destroyed** as soon as Person is destroyed (dead).
- **Aggregation** (City, Tree, Car) "**sub objects**" (Tree, Car) **may NOT be destroyed** when City is destroyed.
- The bottom line is, composition stresses on **mutual existence**, and in aggregation, this property is NOT required.

Read notes section below

Generalization Relationships



- A *generalization* relationship connects a subclass (child class) to its superclass (parent class).
- It denotes an inheritance of attributes and behavior from the superclass to the subclass and indicates a specialization in the subclass of the more general superclass.

Generalisation/Specialisation Relationships

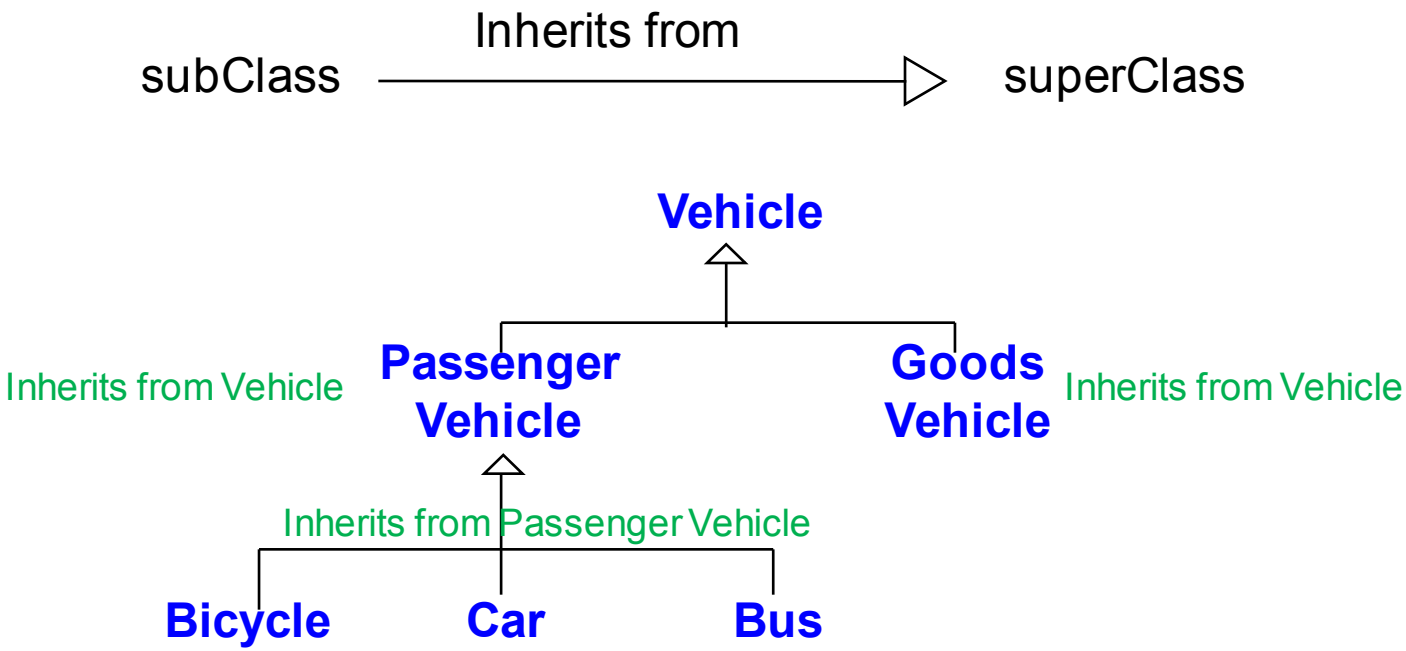
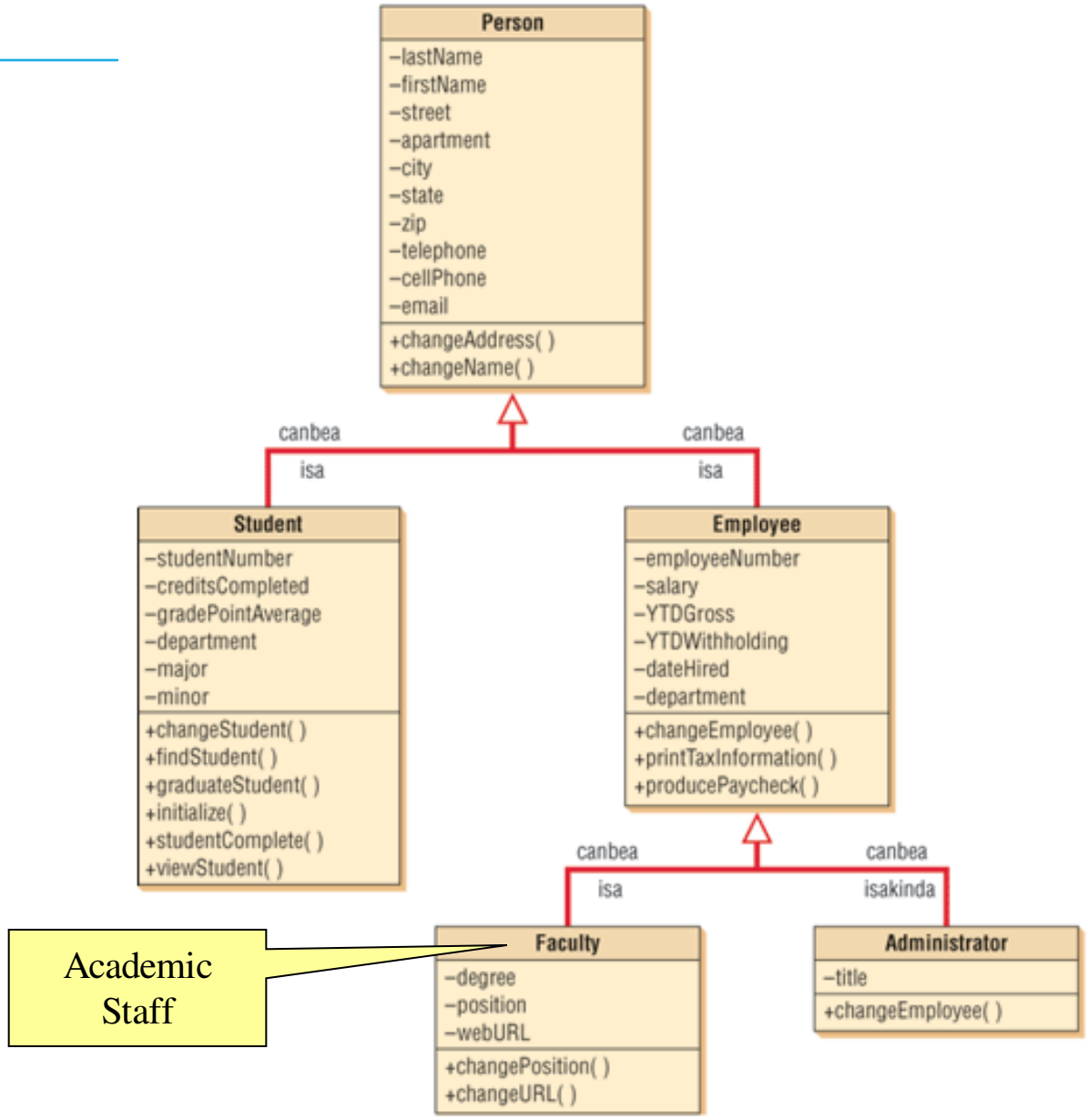
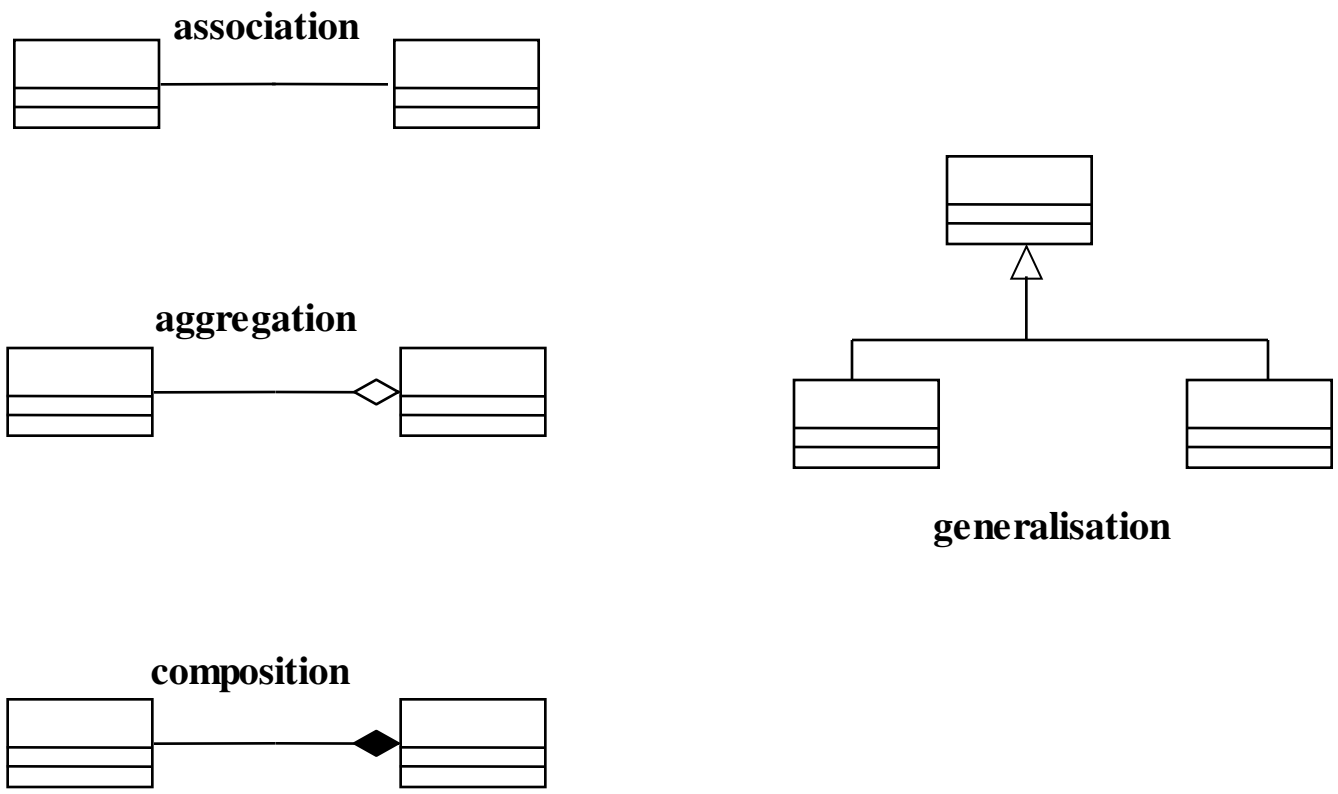


Figure 18.22 A gen/spec diagram is a refined form of a class diagram.



Summary of Relationships between classes



Steps in Developing a Class Diagram

- Identify **Classes**
  - Nouns, concepts, place, people
  - E.g. Subject, Teacher, Administrator, Enrolment
- Identify **Attributes**
  - Possessive phrases; look for the word **"has"**
  - E.g. Student's name, id, code of the course
- Identify **Operations/Methods**
  - Verbs, verbal phrases; look for the word **"can"**
  - E.g. Student can enrol in a Subject
- Identify **Relationships** between classes
  - Association, composition, aggregation and generalisation
- Define **Multiplicities**

What do we have at this stage?

- Use Cases
  - Use case diagram
  - Scenarios/Use Case Narratives

  Agreed by the users or their representatives

Class Diagram: (Where to from here?)

- User Stories (two weeks ago)

- Use Case Diagram (last week)

- Use Case Scenarios or Narratives (last week)

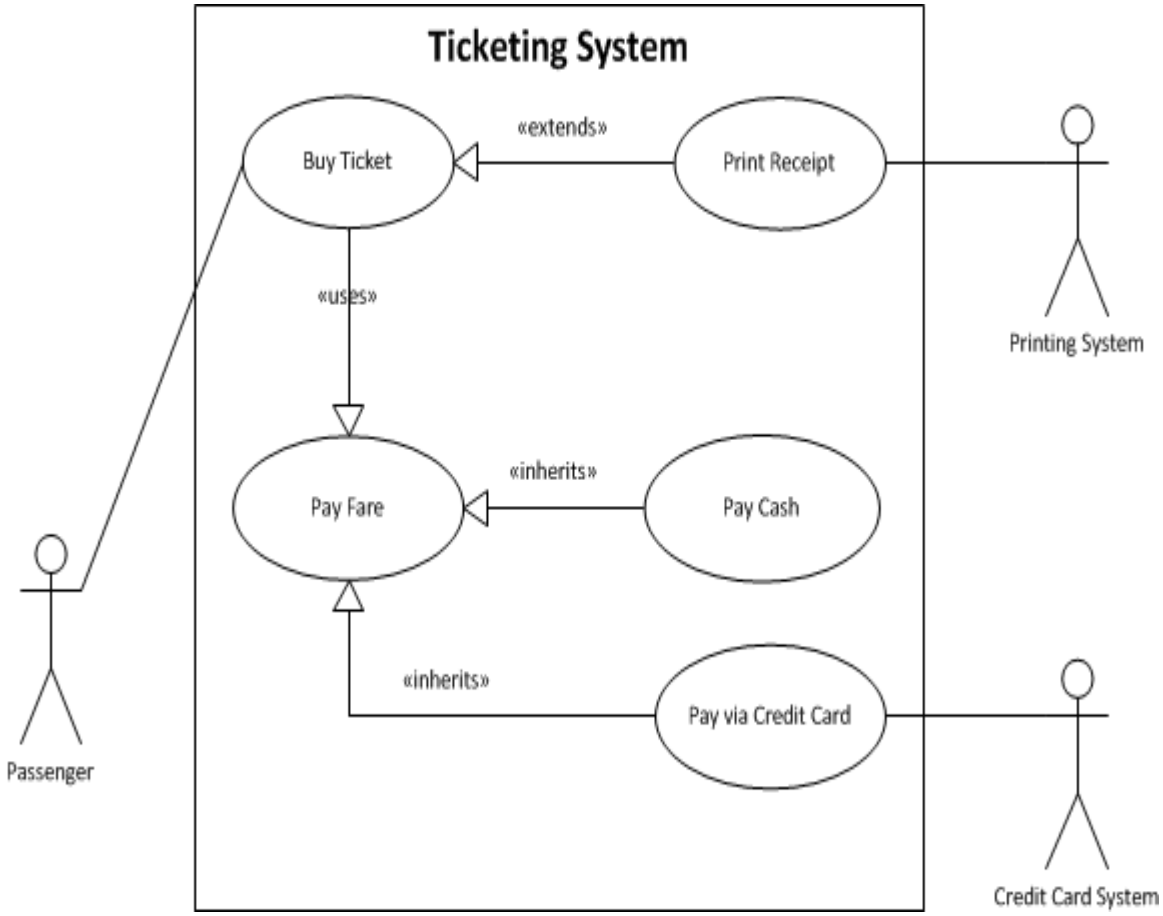
- Class Diagram (one diagram for the whole system)

Read the use case narratives for all use cases and identify Classes and its attributes and methods.

Exercise

- Ticketing System Example from last lecture
- Relate to Assignment 2

Use Case Diagram for Ticketing System



Steps in drawing a Class Diagram for Buy Ticket Use Case Narrative

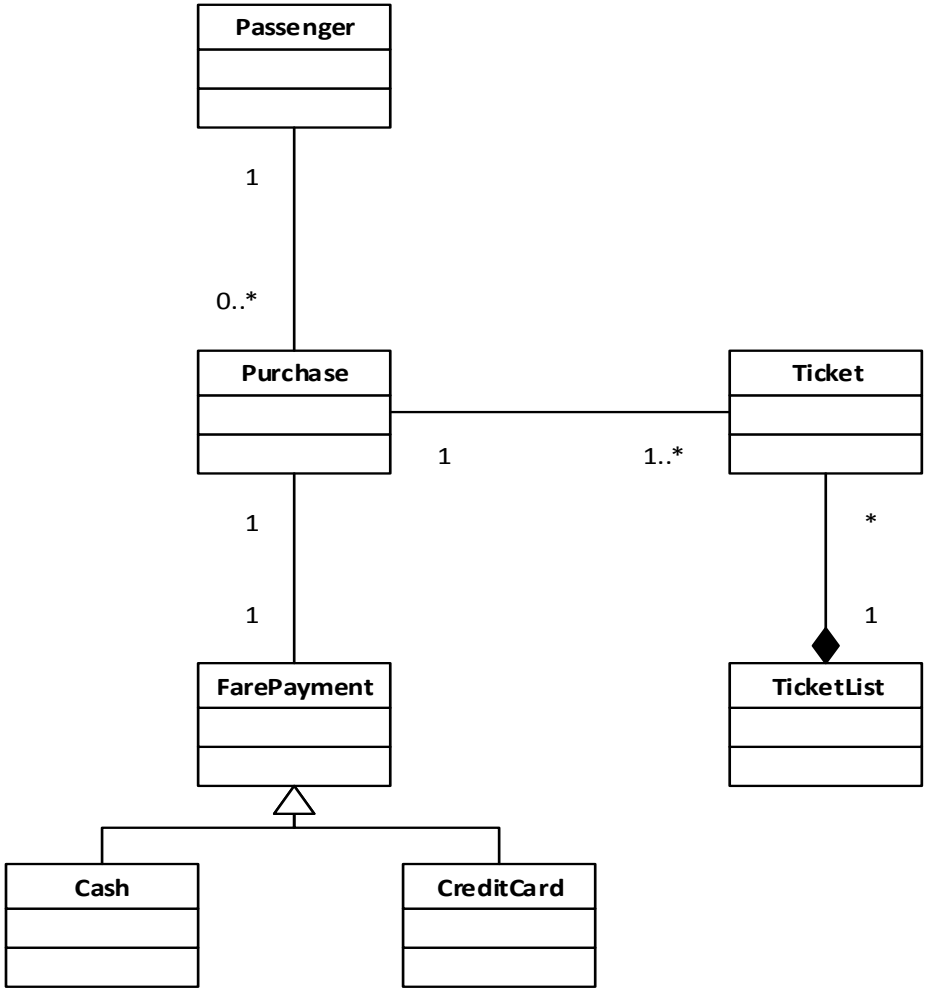
- Refer the Buy Ticket use case narrative from last lecture, available in week 7 folder.
- Read the narrative and follow the following 3 steps:
  - Find Classes/Objects (look for nouns)
  - Identify attributes and methods for each class
  - Identify the relationship and multiplicities between classes.

Buy Ticket Use Case Narrative – from Week 7 Lecture

Buy Ticket Use Case Narrative

Use Case ID	UC101: Buy Ticket
User Story	As a passenger, I want to buy a ticket via Online Ticketing System so that I can travel from one city to another city in Australia.
Goal	Buy a ticket online for traveling from one destination to another destination.
Priority	H
Actors	Primary Actor – Passenger Secondary Actor – Credit Card System, Printing System
Pre-conditions	The passenger has access to the Online Ticketing System. The passenger has a valid credit card.
Post-conditions	The passenger has successfully bought the ticket via the Online Ticketing System.
Trigger	The passenger launches the Online Ticketing (OT) system via their internet browser.
Main Flow	1. The (OT) system displays the OT system landing page and displays the hyperlink to list the available tickets i.e. departure, destination, types, dates and fare. 2. The passenger clicks on the hyperlink to list the available tickets. 3. The OT system displays the available tickets to select from. 4. The passenger selects the desired ticket and presses the "Buy Ticket" button. 5. The OT system displays the **"Pay Fare"** page for the selected ticket and requests for payment. 6. The passenger chooses to "**Pay via Credit Card**". 7. The OT system asks the passenger to enter their credit card details. 8. The passenger submits the credit card details. 9. The OT system checks the credit card number format and processes the credit card details via the **"Credit Card System"**. Please see **"UC102: Pay via Credit Card"** for credit card payment processing details. 10. The OT system successfully receives the payment via the **"Credit**

Exercise – Draft Class Diagram for Ticketing System



Homework: Identify the missing associations, attributes and methods for each class in the above diagram.

Summary

- Classes
- Types
- Relationships
- Application Example

- ***Assignment 2 Hint: Convert ERD into Class Diagram***
- ***So far you should have finished your User Stories and Use Case Narratives for assignment 2.***
- ***Read the narratives and identify Classes and its attributes and methods.***

Assignment 2 – Released Now

- Object Oriented Requirements Analysis and Specification Report – 18 Marks
- Same Case Study as Assignment 1: Customer Onboarding System (COS) for Epic Video
- Functional and non-functional requirements
  - Functional requirements using:
    - User Story Map
    - User Stories and Use Cases (narratives)
    - Sequence Diagram
  - Data Requirements using:
    - Class Diagram
    - State Transition Diagram
  - Non-functional requirements:
    - User Interface requirements using wireframes
    - Security requirements
    - Performance requirements

Assignment 2 Template: Template adapted in this subject

Conclusion

- This Week's Workshop
  - **Quiz 6 – Use Case Modelling (3 marks)** and
  - Tasks – Class Modelling

- Next Week's Lecture
  - Interaction Modelling

- Next Week's Workshop
  - **Quiz 7 – Class Modelling (3 marks)** and
  - Tasks – Interaction Modelling