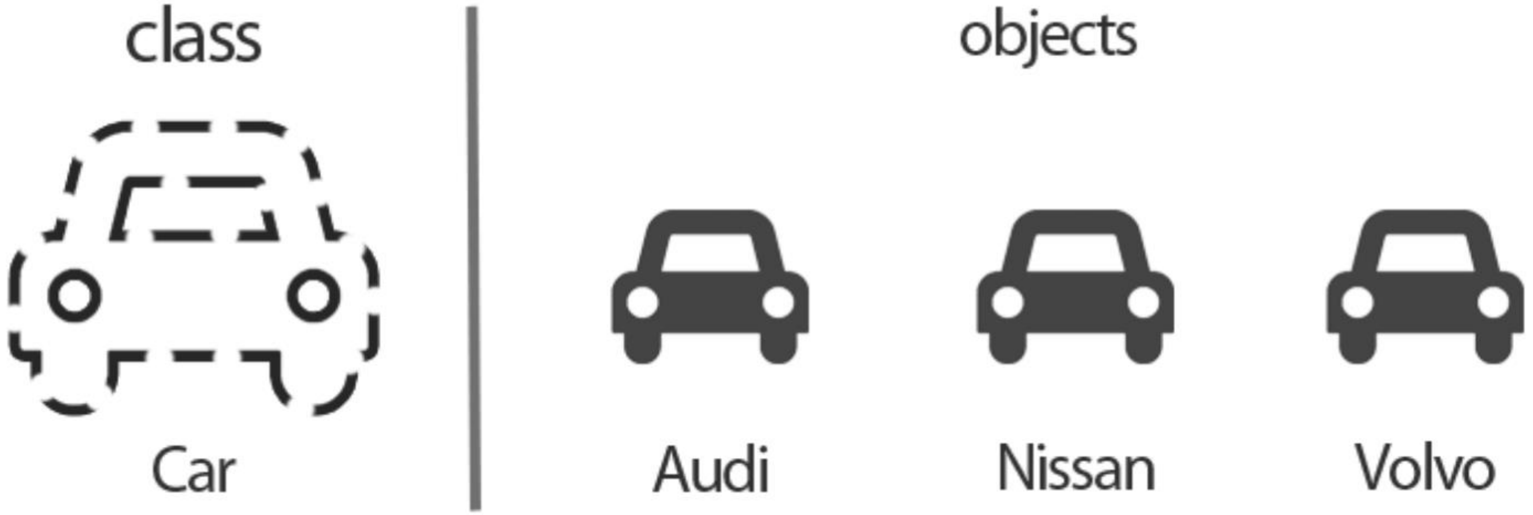
JOHNS HOPKINS
UNIVERSITY

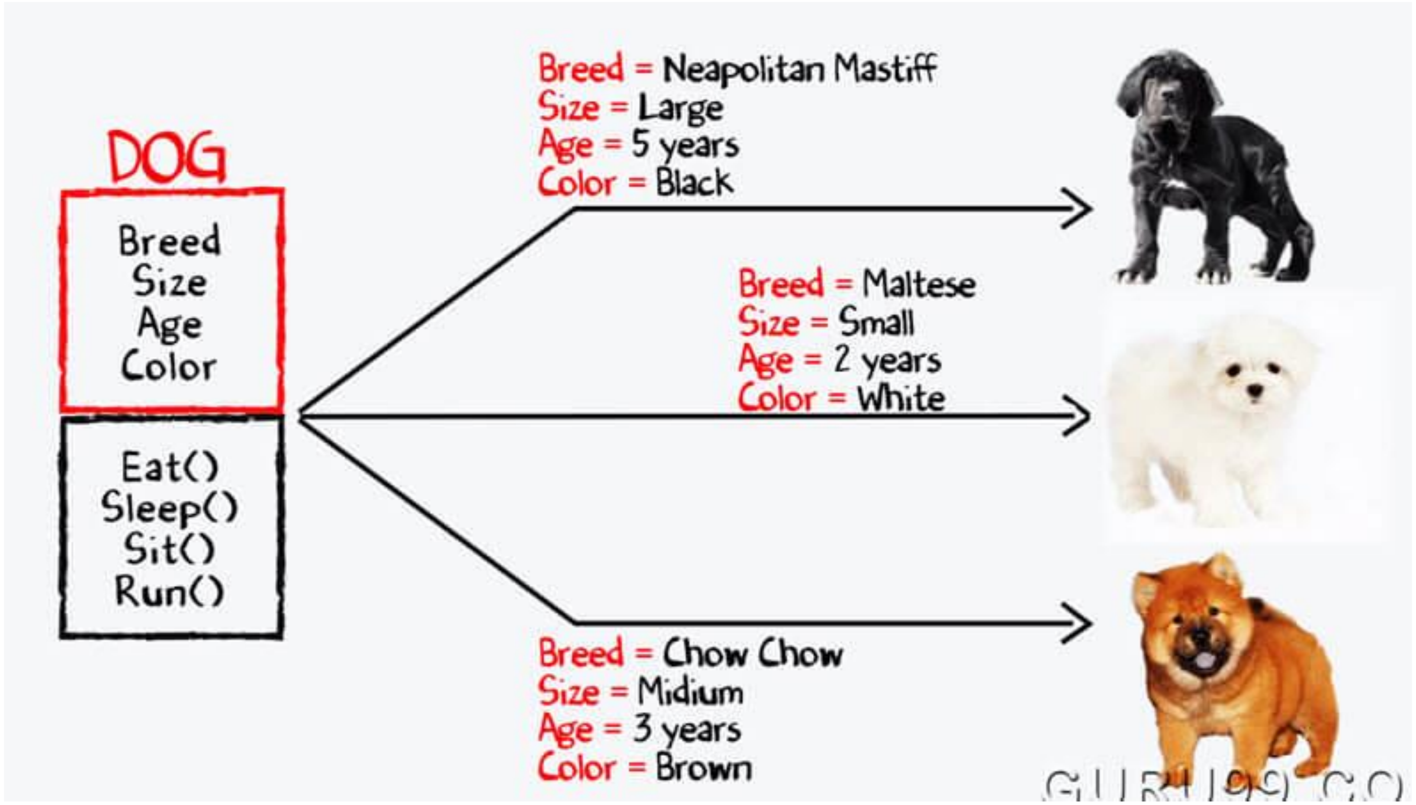
EN.601.421 / EN.601.621

Object Oriented Software Engineering

Class vs. Object

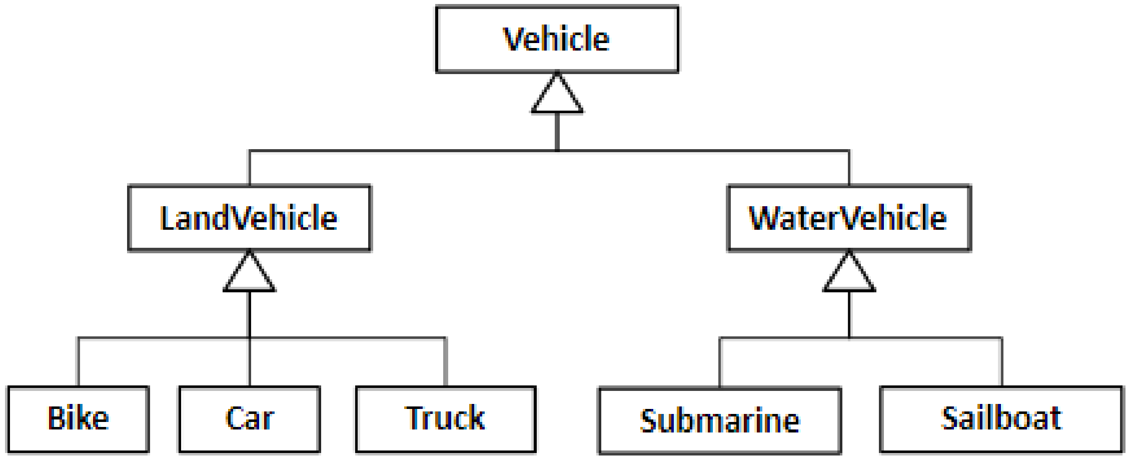


State and Behavior



Encapsulation

Inheritance (generalization)

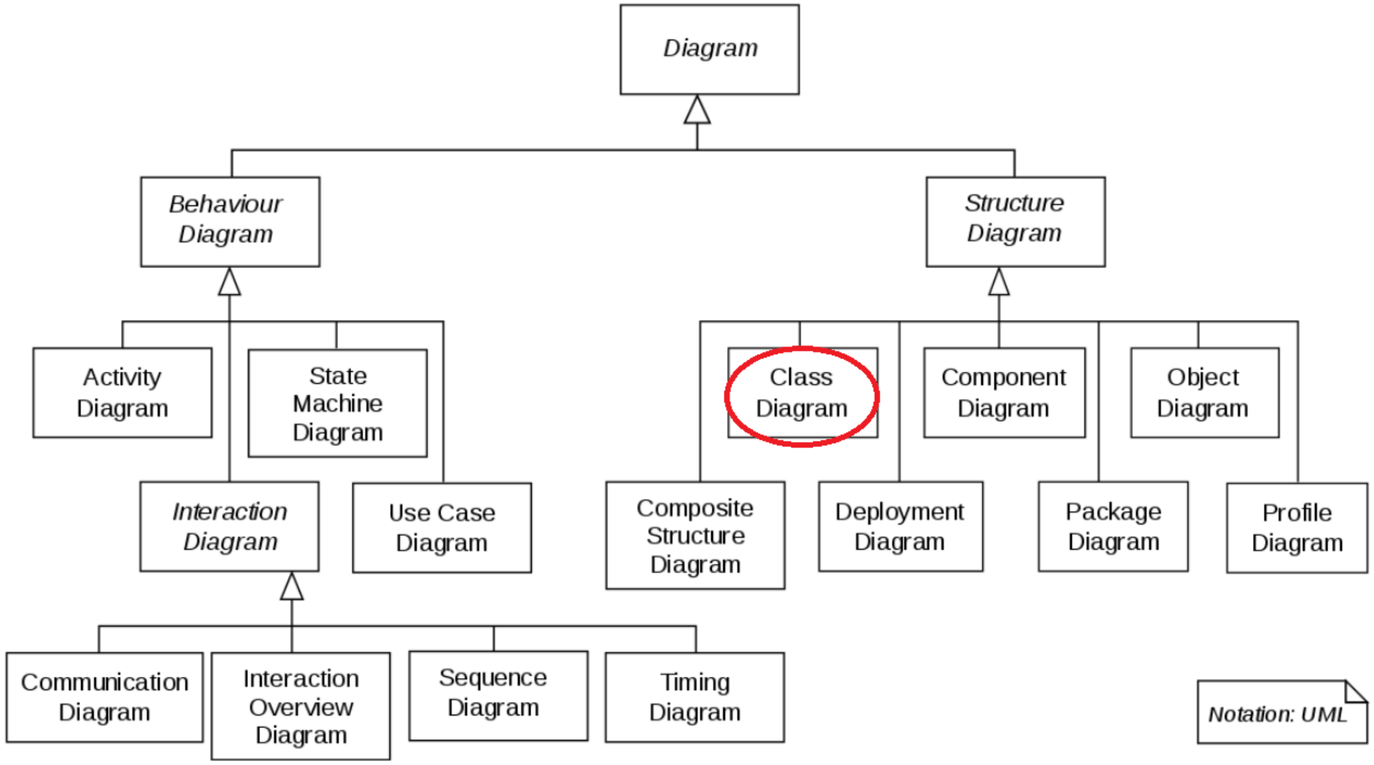


Polymorphism (many forms)

- One Interface – Multiple Implementation

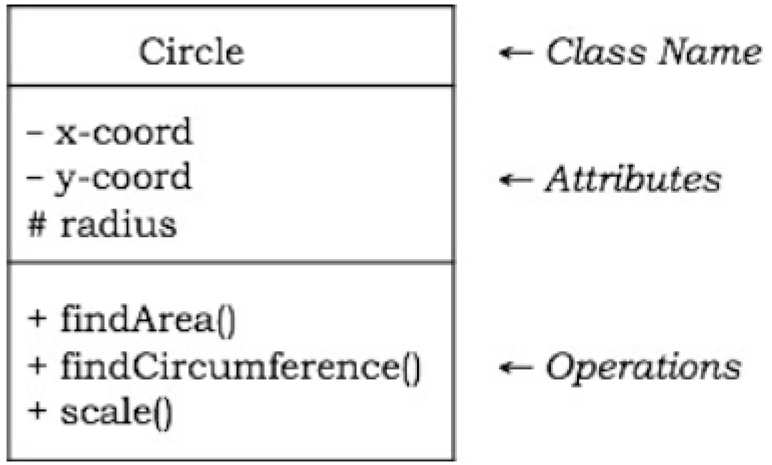
```
public class VehicleCollection {
    private List<Vehicle> vehicles;
    public add (Vehicle v) {
        vehicles.add(v);
    }
    public applyAllBrakes() {
        for (Vehicle v: vehicles) {
            v.applyBrake();
        }
    }
}
```

Unified Modeling Language (UML)

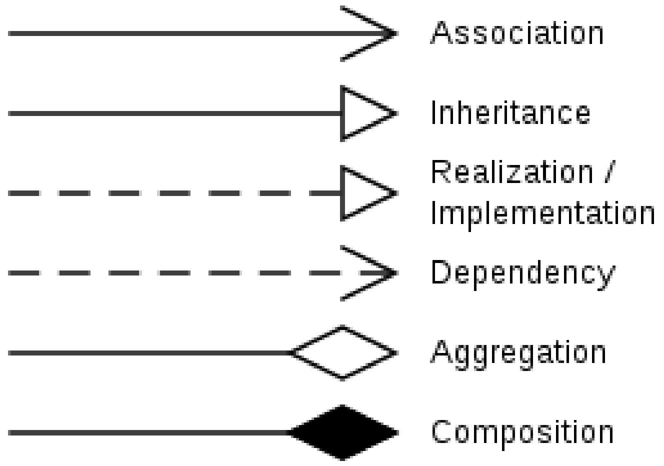


UML

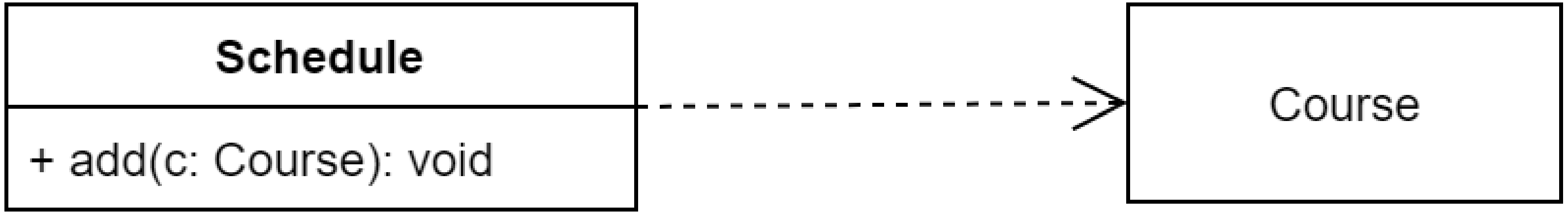
A class is represented as a box

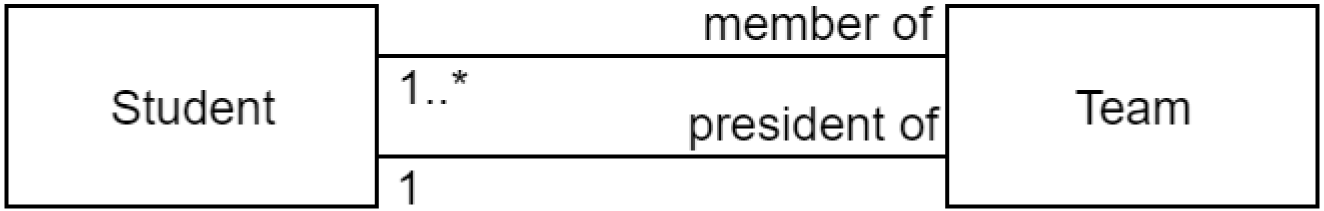


Relationships



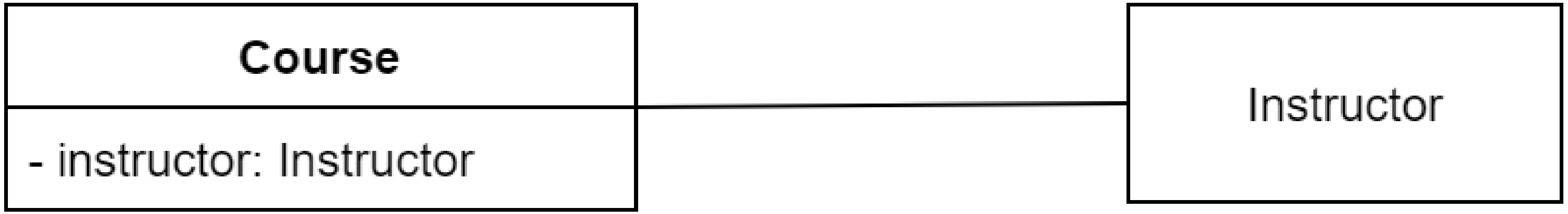
UML Relationships: *Dependency*





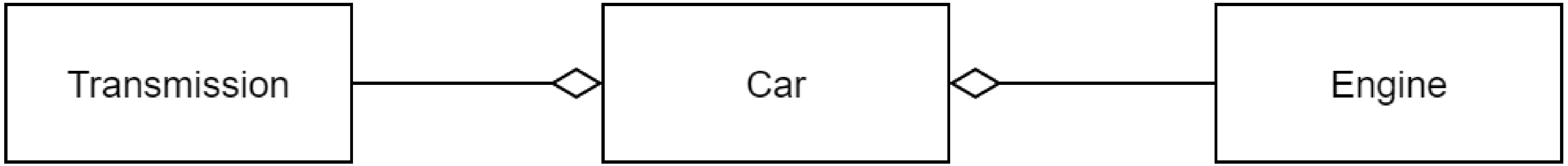
UML Relationships: *Association*

- "Has-a" relationship



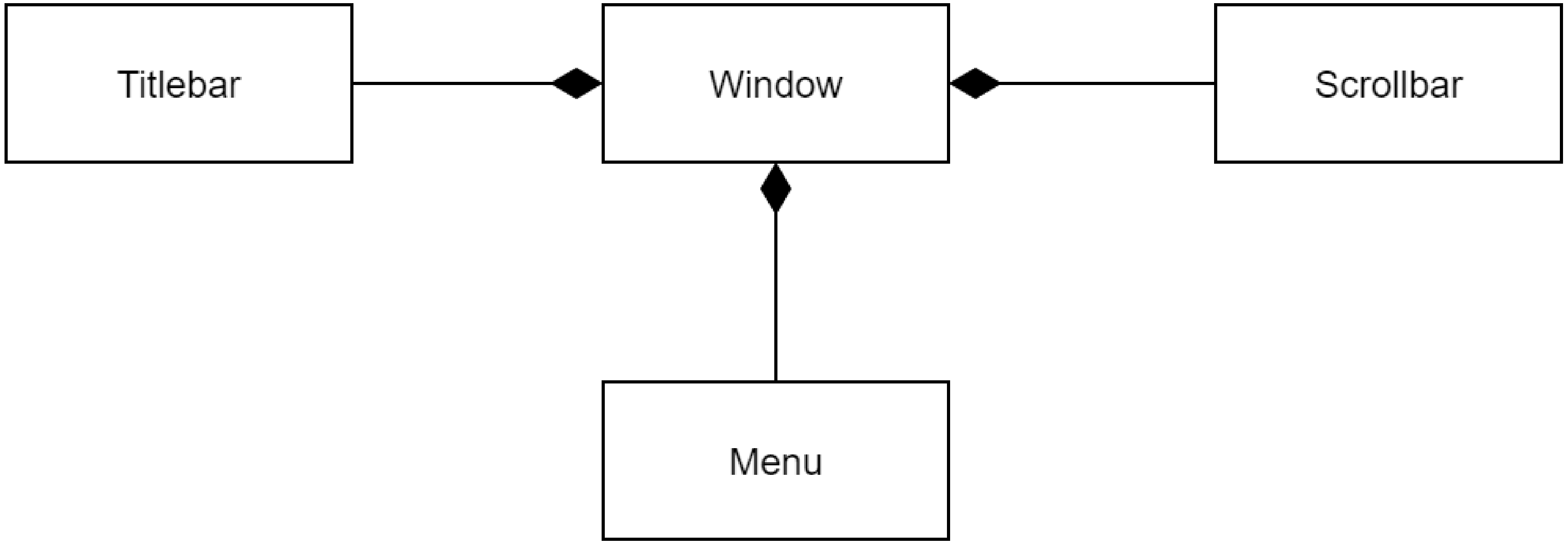
UML Relationships: Aggregation

- Whole-part relationship



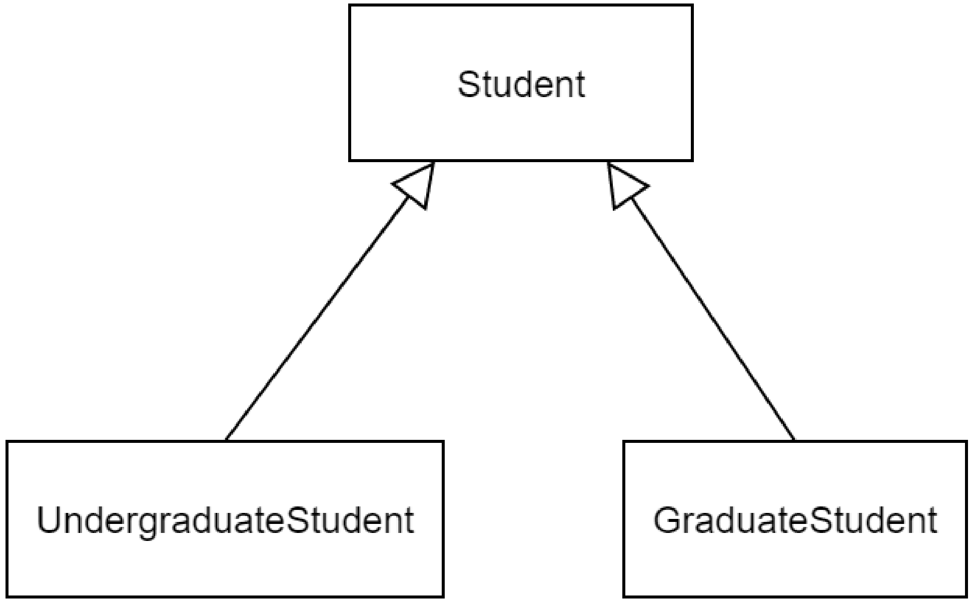
UML Relationships: Composition

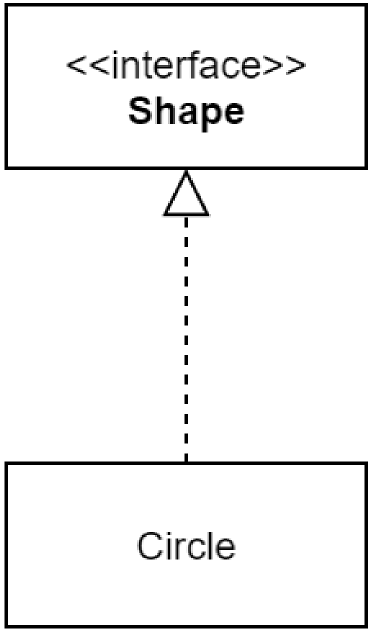
- Strong Ownership



UML Relationships: Generalization

- Inheritance





Object-Oriented Analysis & Design (OOAD)

- Model a system as a group of interacting objects:
  - **Analysis**, or *problem modeling*, in which the problem is described and represented.
  - **Design**, or *solution modeling*, in which a solution to the problem is discovered and represented.
  - **Implementation**, in which the code that makes up the working system is written and tested.

Object-Oriented Analysis & Design (OOAD)

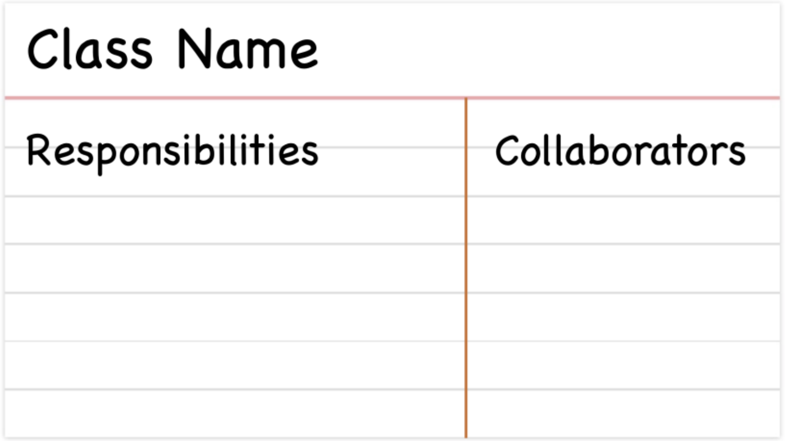
- After writing SRS you *identify classes* to:
  1. Model your application
  2. Easy to change

Identify Classes

- The verb-noun technique can learn about existing vacancies.

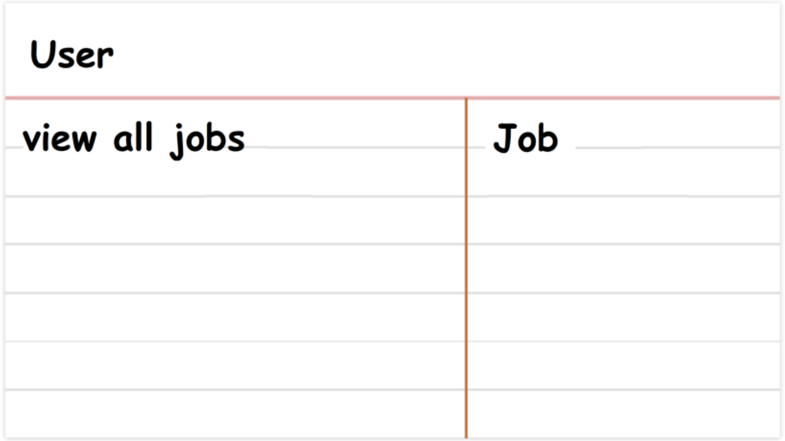
"As a user, I want to view a complete list of all posted jobs so that I can learn about existing vacancies."

Class Responsibility Collaborator (CRC)



Class Name

Responsibilities	Collaborators

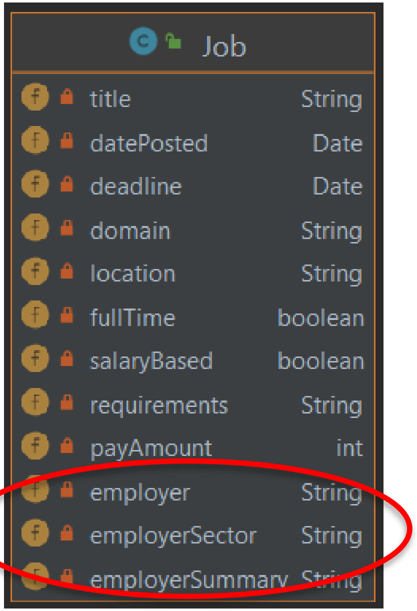


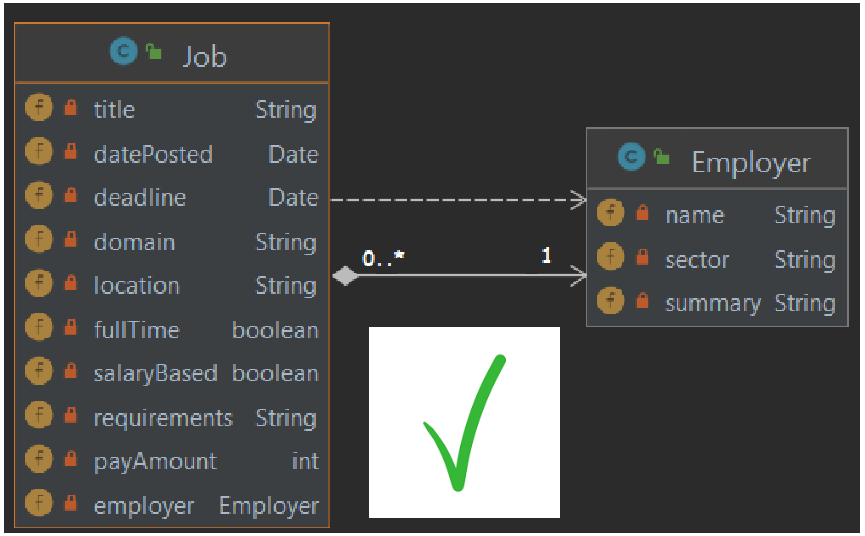
User

view all jobs	Job

Increase Cohesion

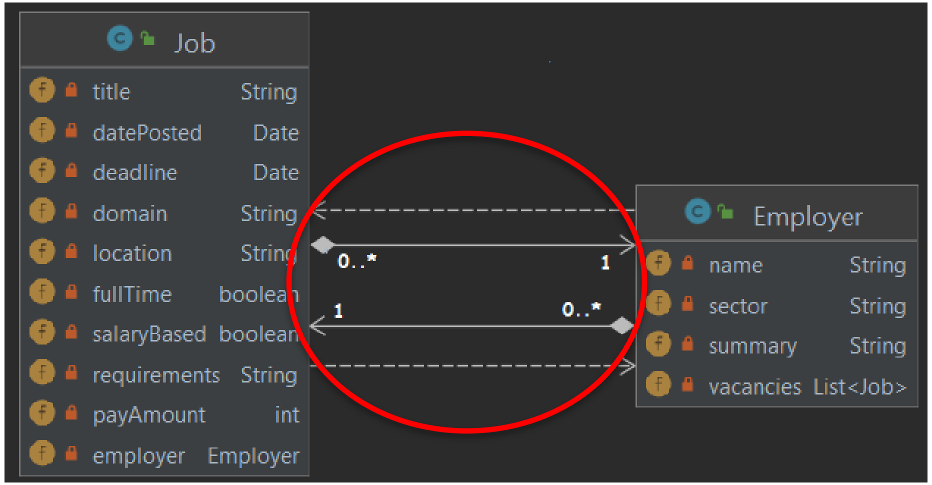
- A highly cohesive class is one that only comprises responsibilities which belong together. A class ideally has a single responsibility.

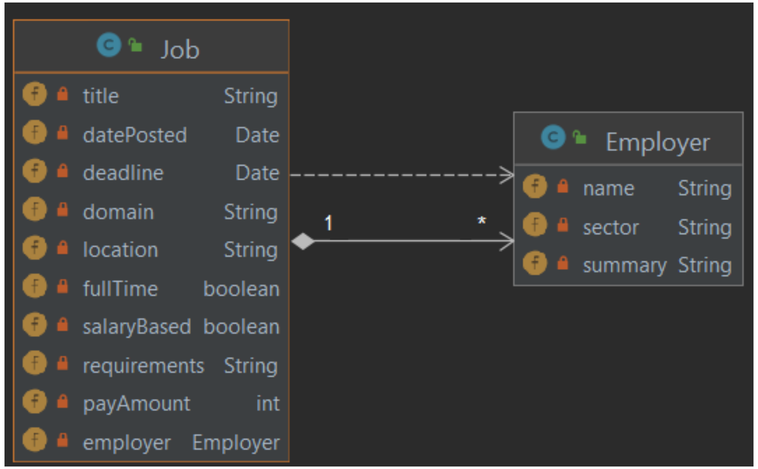




Decrease Coupling

- A class should not interact (collaborate) with too many other classes. If it does, it should be *loose*





Loose Coupling

```
public class Student {
   private String name;
   private String email;
   private GradeBook grades;

   public double getScore() {
      double quiz = grades.quiz();
      double project = 0;
      for (Double iteration: grades.project()) {
         project += iteration;
      }

      double homework = 0;
      for (Double grade: grades.homework()) {
         homework += grade;
      }
      return 0.1 * quiz + 0.3 * homework + 0.6 * project;
   }
}
```

✗

```
public class Student {
  private String name;
  private String email;
  private GradeBook grades;

  public double getScore() {
    return grades.totalScore();
  }
}
```

✓

JOHNS HOPKINS

WHITING SCHOOL
of ENGINEERING