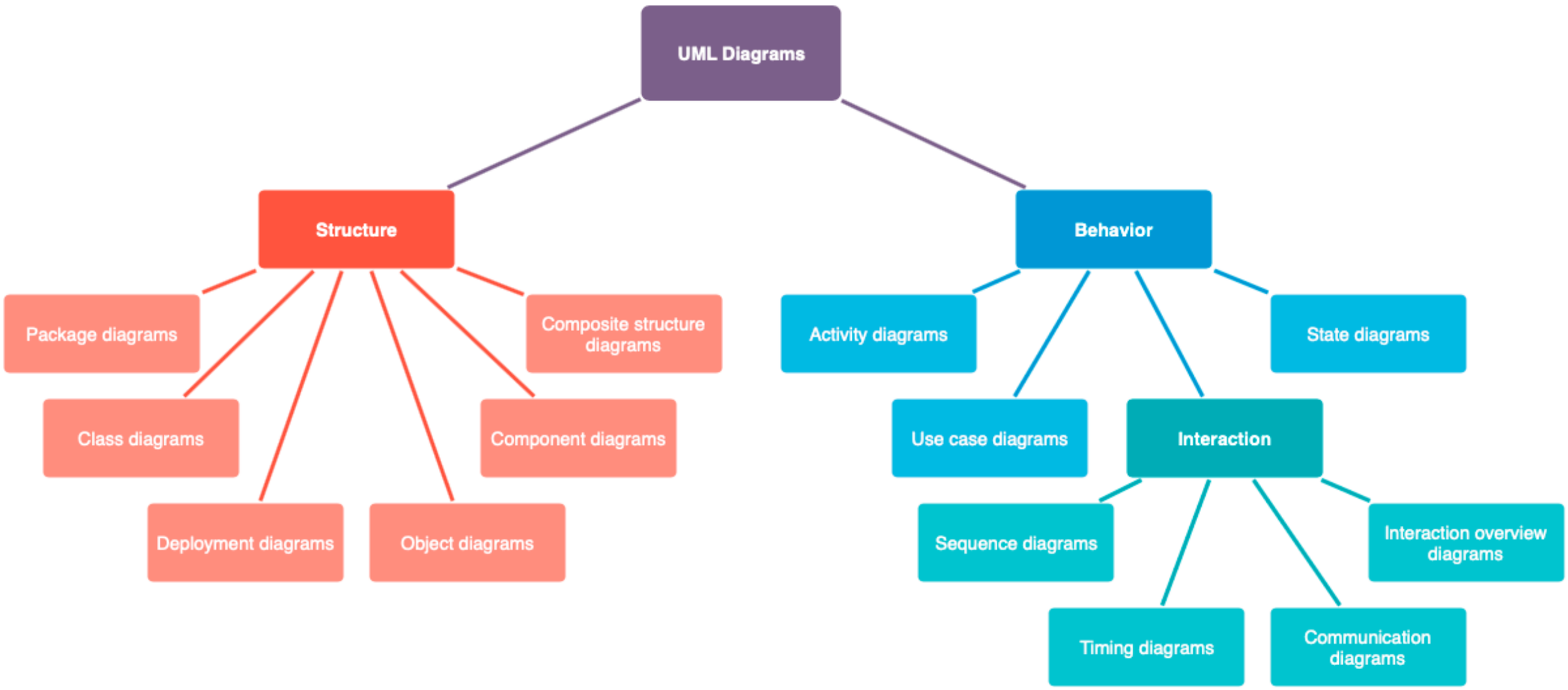


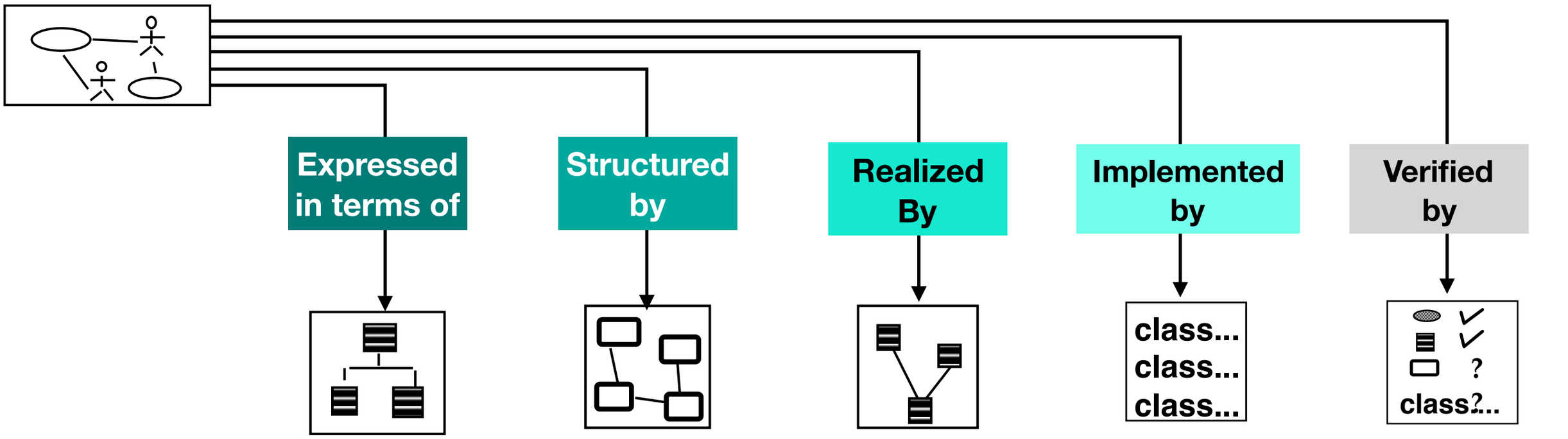
Class Diagram

Jin Hyun Kim

Fall, 2019

Review

Requirement Elicitation	Analysis	System Design	Detailed Design	Implementation	Testing



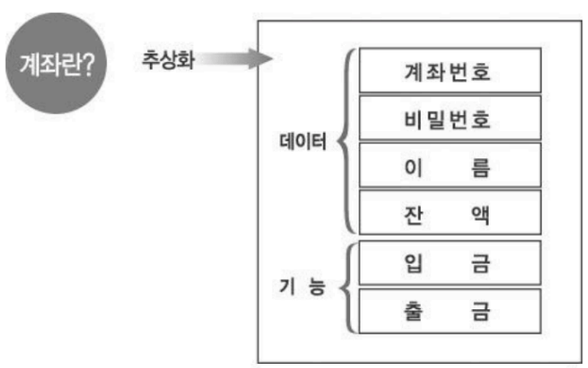
Use-Case Model	Application-Domain Model	Sub-system	Solution-Domain Model	Source Code	Test Case

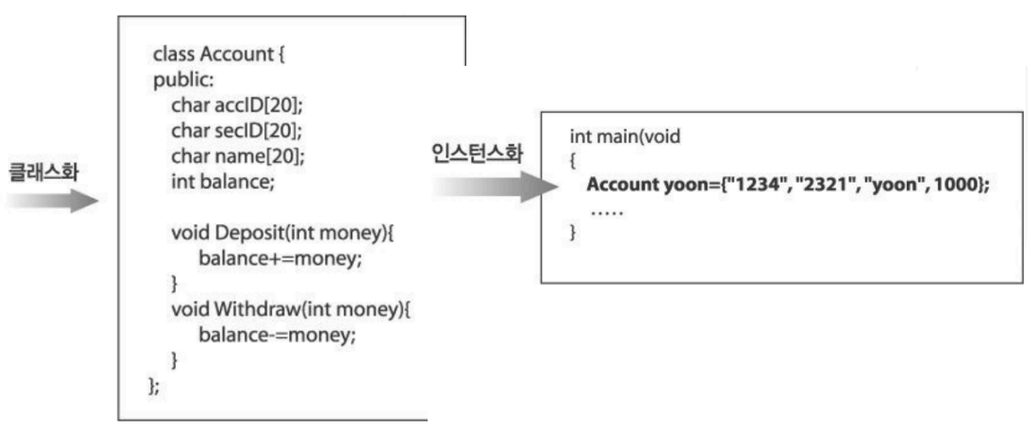
Review

Requirement Elicitation	Analysis	System Design	Detailed Design	Implementation	Testing
Use-Case Model	Application-Domain Model	Sub-system	Solution-Domain Model	Source Code	Test Case

Abstraction → Class Diagram → Sequence Diagram → Program (Objects)

Derive Objects → Class Diagram





Class Diagram

- Describe the structure of the system (시스템의 구조를 모델링)

- Primary purpose during analysis workflow:

  - to create a ***vocabulary, e.g., variables and operators,*** that is used by both the analyst and users (사용자와 분석자에 의해서 사용되는 어휘를 만들기 위환 절차)

Class

- A general **template** that we use to create specific instances or objects in the application domain

- Represents a kind of person, place, or thing about which the system will need to capture and store information

- Abstractions that specify the ***attributes*** and ***behaviors*** of a set of objects

Object

- Entities that encapsulate ***state*** and ***behavior***
- Each object has an identity
  - It can be referred individually
  - It is distinguishable from other objects

Types of Classes

- Ones found during analysis:
    - People, places, events, and things about which the system will capture information
        - Ones found in application domain
- Ones found during design
    - Specific objects like windows and forms that are used to build the system

Classes in Analysis

- Concrete
  - Class from application domain
    - Example: Customer class and Employee class
- Abstract
  - Useful abstractions
  - Example: Person class

Class

Attributes
(성질, 특성, 데이터, 상태,...)

Operations
(행위, 메소드, 함수, ...)

Attributes

- Properties of the class about which we want to capture information (우리가 알(프로세싱)고자하는 정보의 특성)

- Represents a piece of information that is relevant to the description of the class within the application domain (애플케이션 도메인 내의 클래스에 대한 설명과 관련된 정보를 표현)

Attributes

- Only add attributes that are primitive or atomic types
- Derived attribute
    - Attributes that are calculated or derived from other attributes

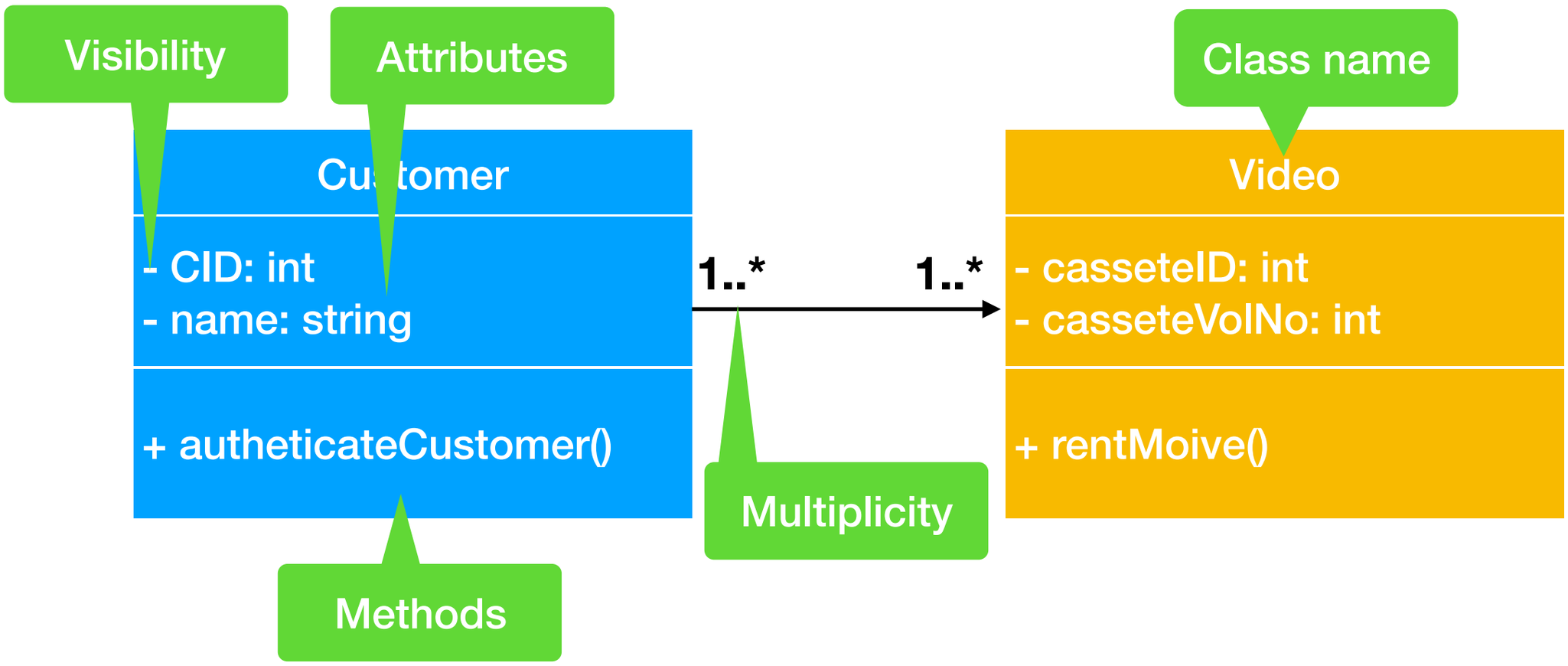
Operations

- Represents the actions or functions that a class can perform

- Describes the actions to which the instances of the class will be capable of responding

- Can be classified as a constructor, query, or update operation

Example of Class Diagram (CD)



Visibility

- Relates to the level of information hiding to be enforced

Visibility	Symbol	Accesible to
Public	+	All objects within your system
Protected	#	Instances of the implementing class and its subclasses.
Private	-	Instances of the implementing class.

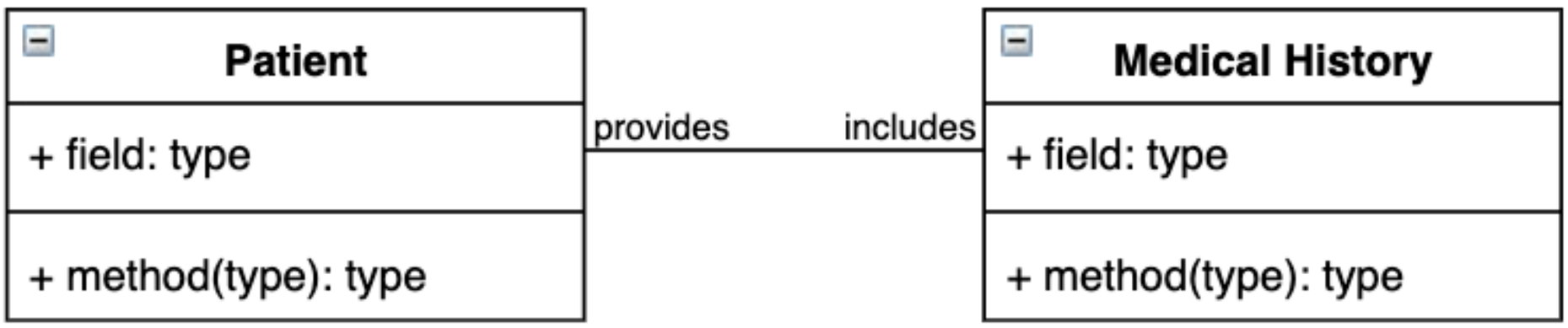
Relationships among Classes

- Represents a connection between multiple classes or a class and itself

- 3 basic categories:
  - Generalization
  - Association
  - Aggregation
  - Composition

Association

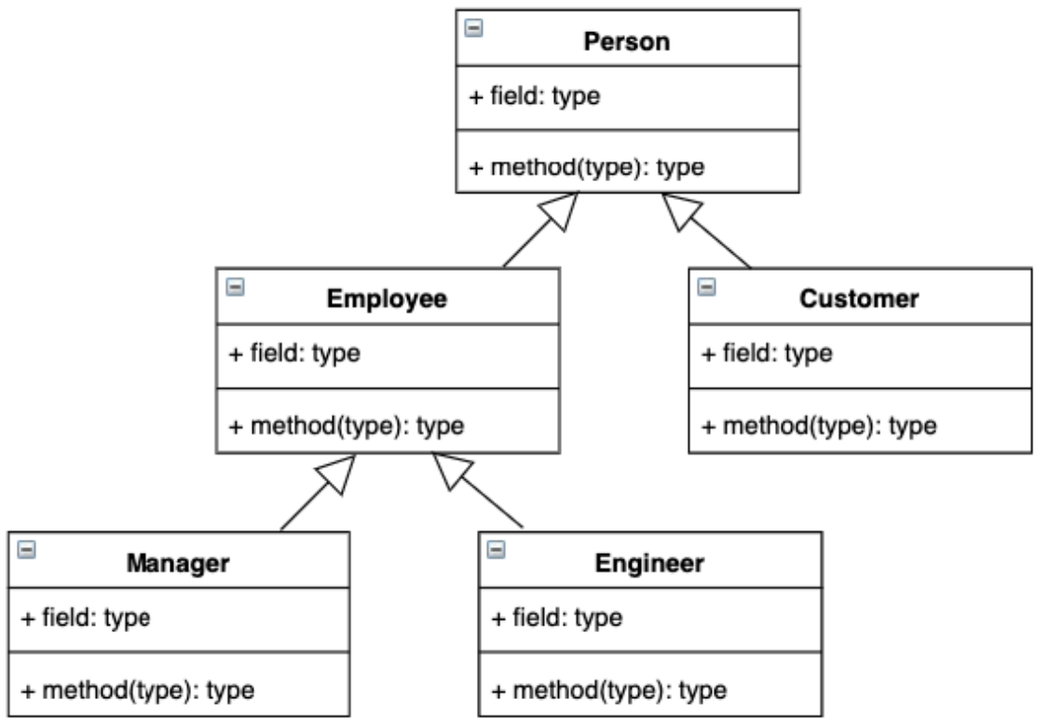
- Name indicates how two classes are related to each other



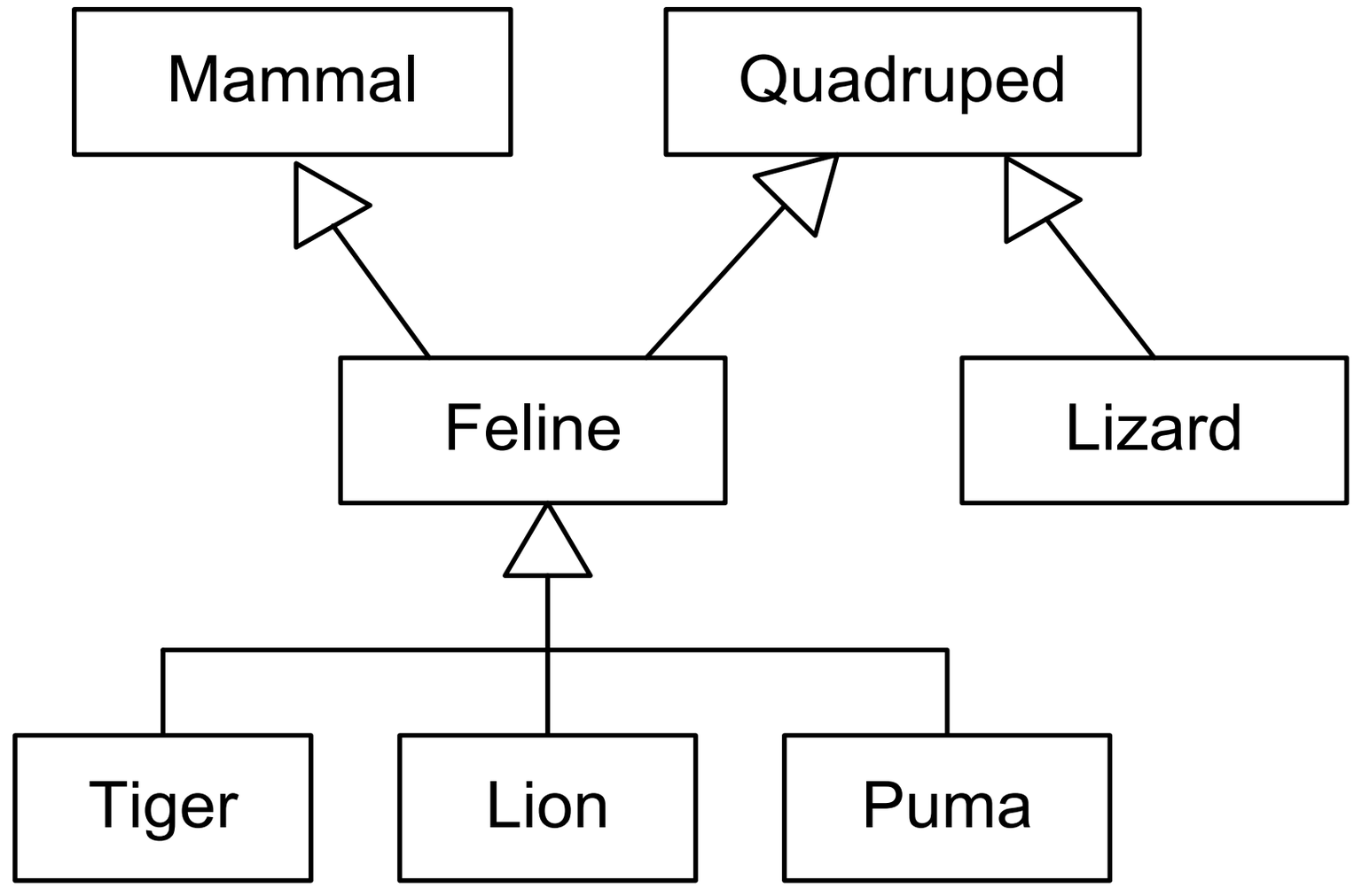
Generalization

- Enables the analyst to create classes that inherit attributes and operations of other classes

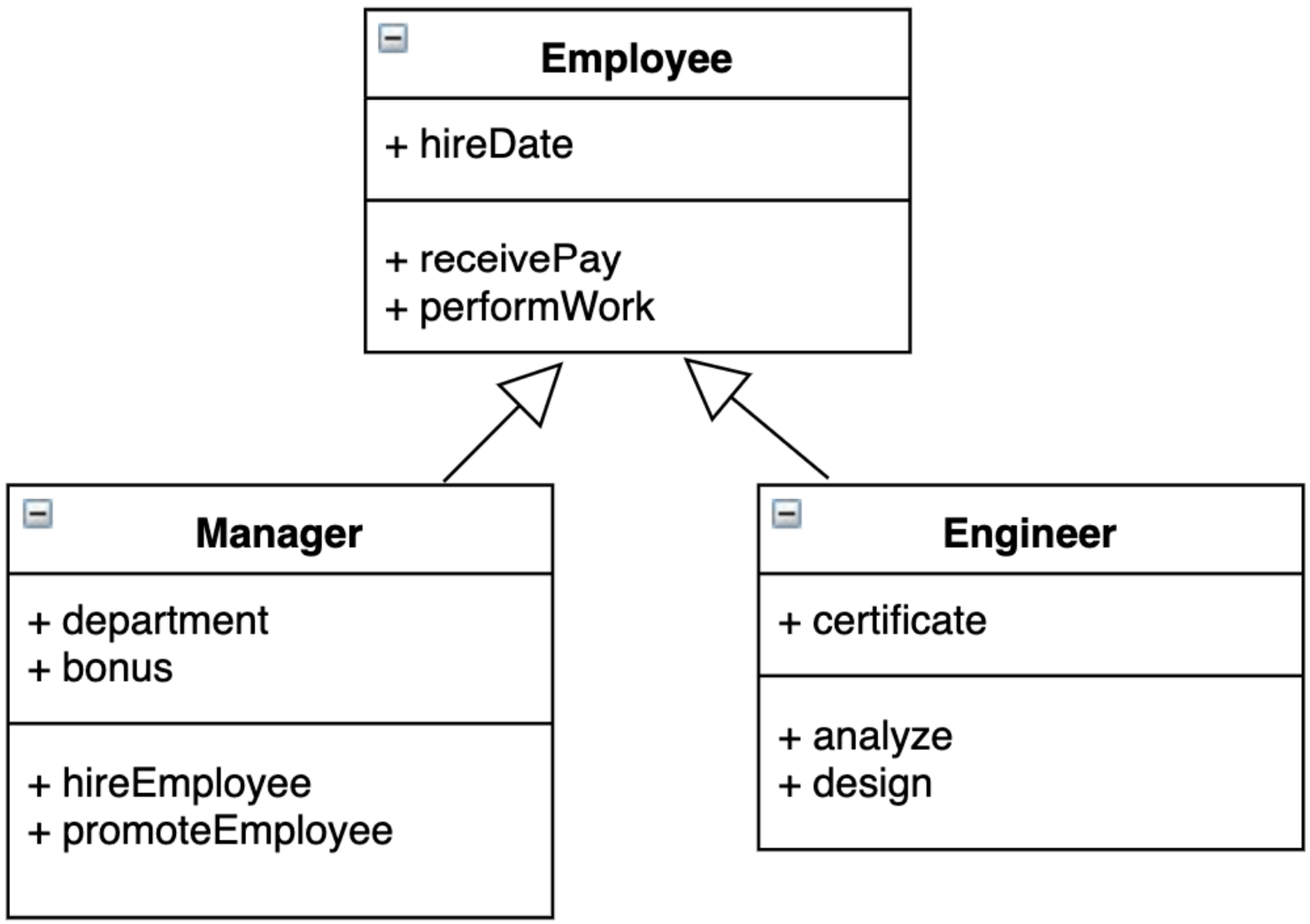
- Represented by a-kind-of relationship



Generalization

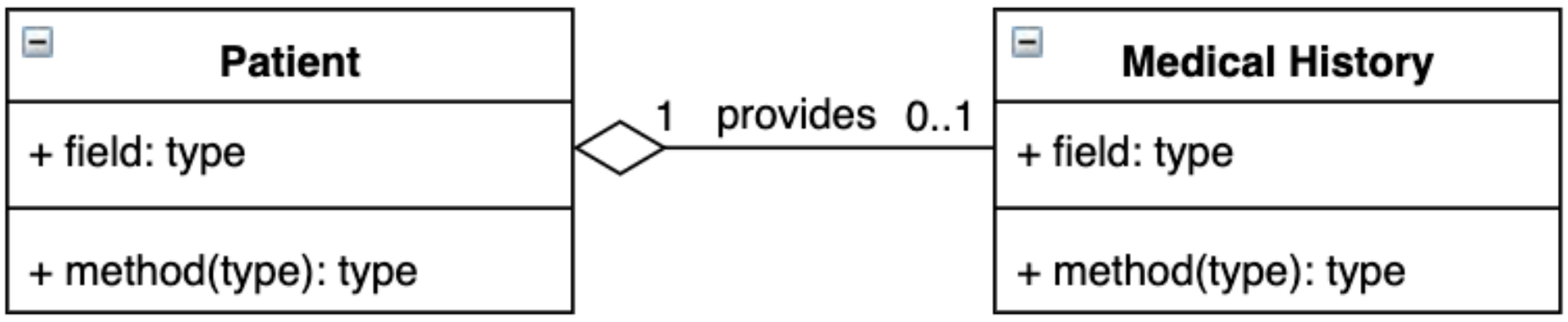


Example


Employee
+ hireDate
+ receivePay
+ performWork
Manager
+ department
+ bonus
+ hireEmployee
+ promoteEmployee
Engineer
+ certificate
+ analyze
+ design

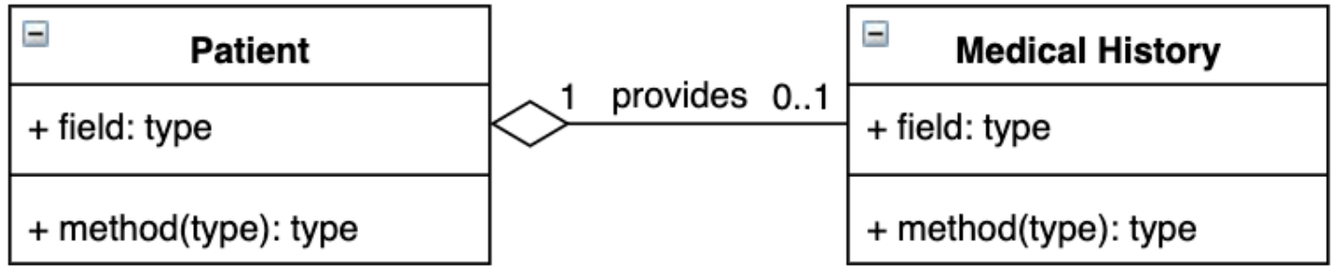
Aggregation

- Specialized form of association in which a whole is related to its part(s)
- Represented by a-part-of relationship (부분 관계)



Aggregation

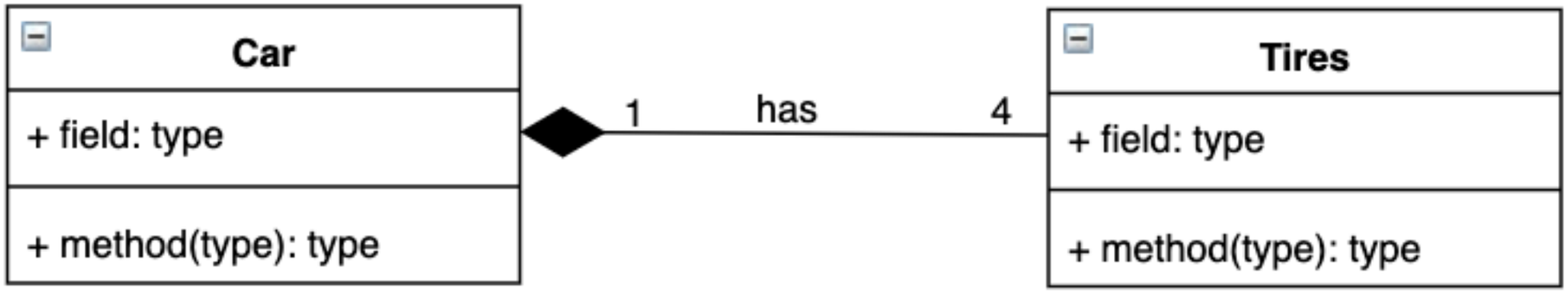
- Multiplicity: Documents how many instances of a class can be associated with one instance of another class (얼마나 많은 클래스의 인스턴스(객체)가 연결되어 있는지 표현)



Type	Expression
Exactly one	1
Zero or more	0..*
One or more	1..*
Zero or one	0..1
Specified range	2..4
Multiple, disjoint ranges	1..3, 5

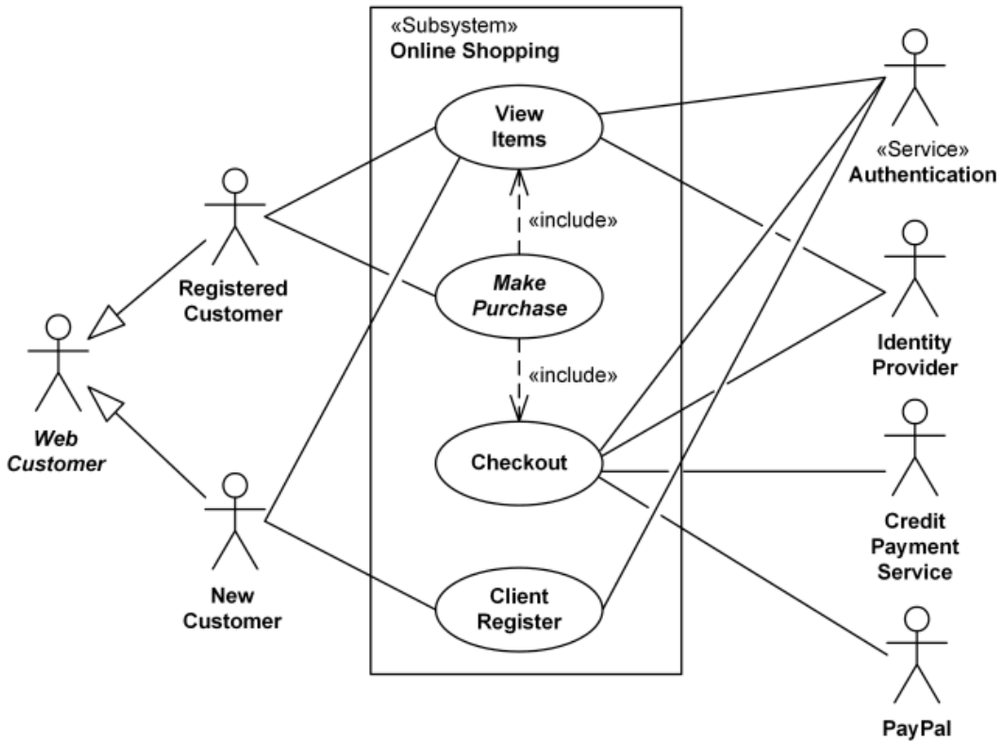
Composition

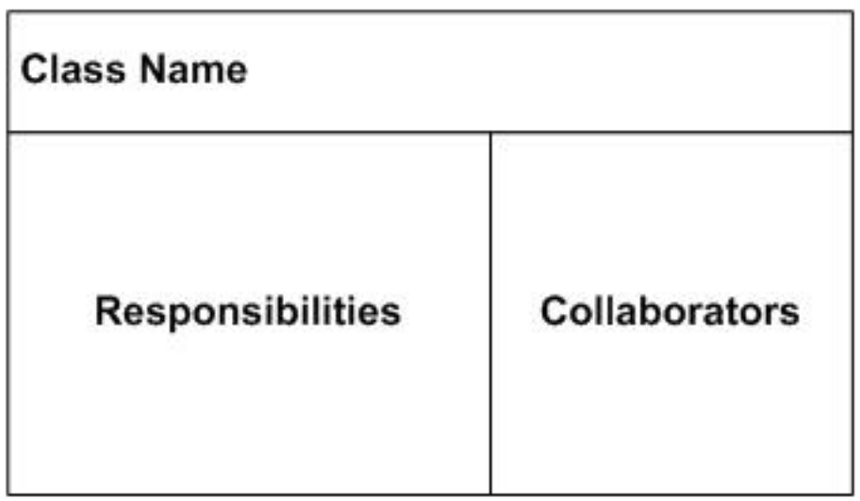
- Strong relationship that two classes depend on each other



CD Derivation

- From UC and CRC (Class, Responsibility, Collaborator) cards





CD Derivation from UC

- From UC and their scenarios
  - Analyze the text in the use-case descriptions and scenarios (UC 명세와 시나리오의 텍스트를 분석)
  - A common and improper noun and implies a class of objects (일반 명사 혹은 비고유명사는 객체의 클래스)
  - A proper noun implies an instance of a class (고유명사는 클래스의 객체)
  - A collective noun implies a class of objects made up of groups of instances of another class (집합 명사는 다른 클래스들의 객체 그룹으로 만들어진 객체)

CD Derivation from UC

- An adjective implies an attribute of an object (형용사는 객체의 특성)
- A doing verb implies an operation (Doing 동사는 연산)
- A being verb implies a relationship between an object and its class (Being 동사는 객체와 그 클래스 간의 관계)
- A having verb implies an aggregation or association relationship (Having-동사는 집합이나 연관관계)

CD Derivation from UC

- A transitive verb implies an operation (능동태 동사는 연산)
- An intransitive verb implies an exception (수동태 동사는 예외)
- A predicate or descriptive verb phrase implies an operation (조건이나 설명 동사구는 연산)
- An adverb implies an attribute of a relationship or an operation (부사는 관계 혹은 연산의 특성)

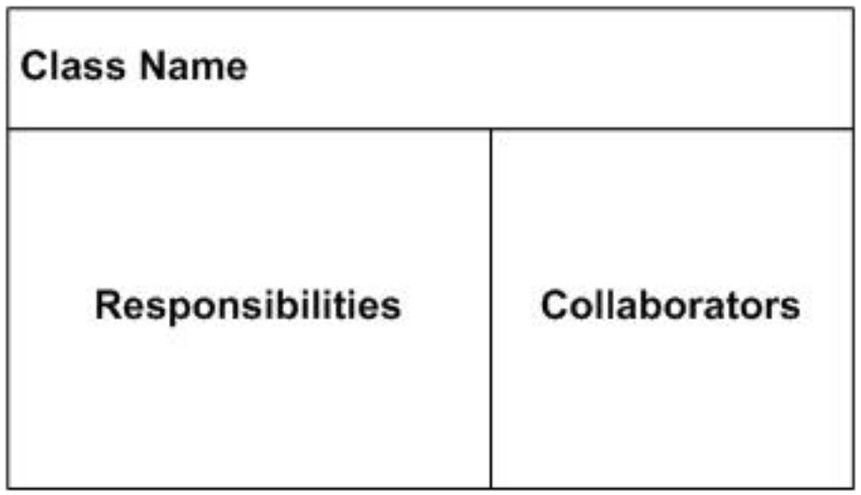
Example

Part of speech	model component	Examples
Proper noun	Instance (object)	Alice, Ace of Hearts
Common noun	Class (or attribute)	Field Officer, PlayingCard, value
Doing verb	Operation	Creates, submits, shuffles
Being verb	Inheritance	Is a kind of, is one of either
Having verb	Aggregation/Composition	Has, consists of, includes
Modal verb	Constraint	Must be
Adjective	Helps identify an attribute	a *yellow* ball (i.e. color)

Principles in using CD

- Ensure that the classes are both necessary and sufficient to solve the underlying problem
  - no missing attributes or methods in each class
  - no extra or unused attributes or methods in each class
  - no missing or extra classes

CRC Cards

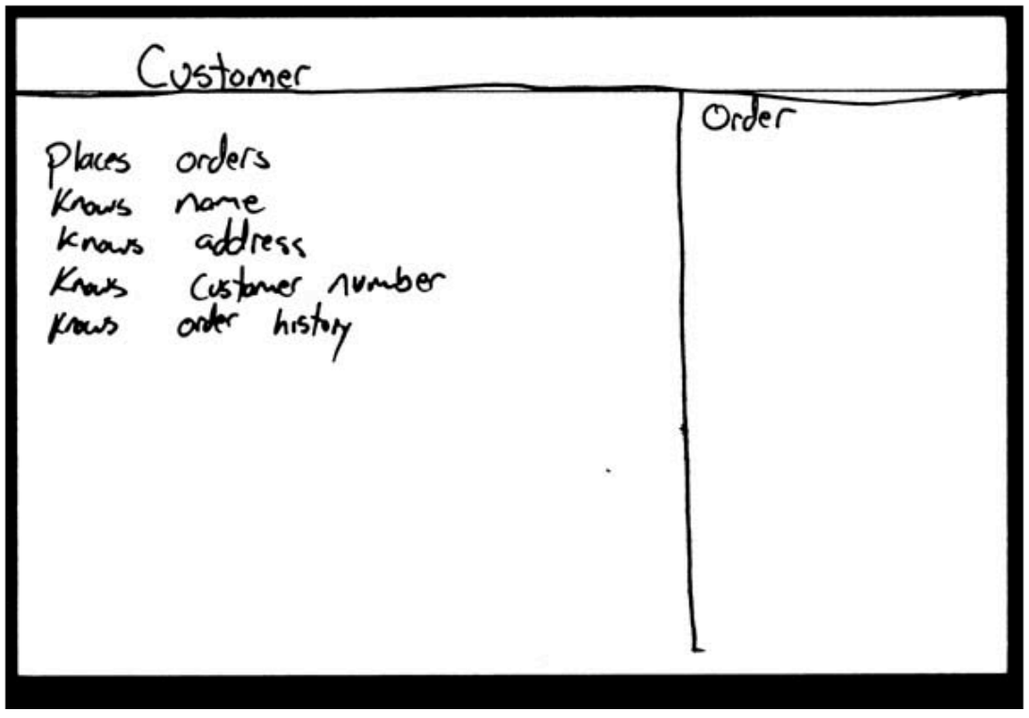


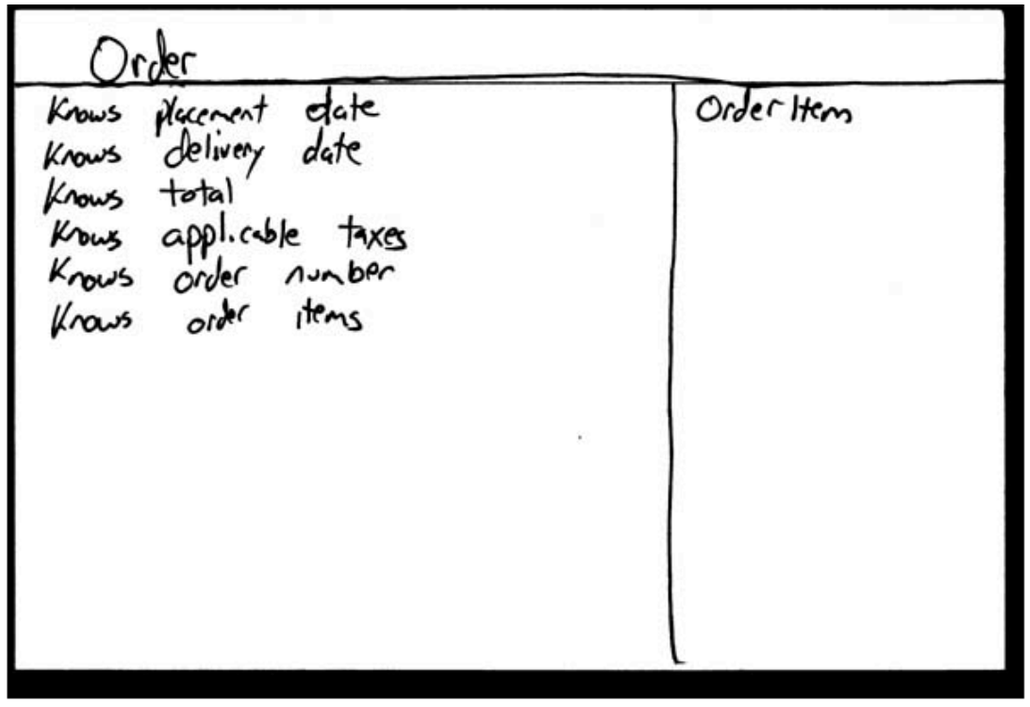
- A class represents a collection of similar objects (클래스는 비슷한 객체들의 모임)

- A responsibility is something that a class knows or does (책임은 클래스가 알거나 하는 것들)

- A collaborator is another class that a class interacts with to fulfill its responsibilities. (동업자는 다른 클래스로 어떤 클래스가 상호작용함으로 그의 책임을 완수함.)

Hand-Written CRC





Example

Student	
Student number Name Address Phone number Enroll in a seminar Drop a seminar Request transcripts	Seminar

- The card Student requests an indication from the card Seminar whether a space is available, a request for information (학생은 세미나 카드로부터 세미나에 빈 자리가 있는지를 확인할 수 있는 정보를 요청-요청할 정보)

Example

Student	
Student number Name Address Phone number Enroll in a seminar Drop a seminar Request transcripts	Seminar

- Student then requests to be added to the Seminar, a request to do something (그리고 나서 세미나에 등록을 요청 - 해야할 일)

Another Example

Enrollment	
Mark(s) received Average to date Final grade Student Seminar	Seminar

Transcript	
See the prototype Determine average mark	Student Seminar Professor Enrollment

Student Schedule	
See the prototype	Seminar Professor Student Enrollment Room

Room	
Building Room number Type (Lab, class, ...) Number of Seats Get building name Provide available time slots	Building

Professor	
Name Address Phone number Email address Salary Provide information Seminars instructing	Seminar

Seminar	
Name Seminar number Fees Waiting list Enrolled students Instructor Add student Drop student	Student Professor

Student	
Name Address Phone number Email address Student number Average mark received Validate identifying info Provide list of seminars taken	Enrollment

Building	
Building Name Rooms Provide name Provide list of available rooms for a given time period	Room

Exercise

- For given UC diagram, create scenarios for each uses case

- Based on scenarios, create CD using the technique above.