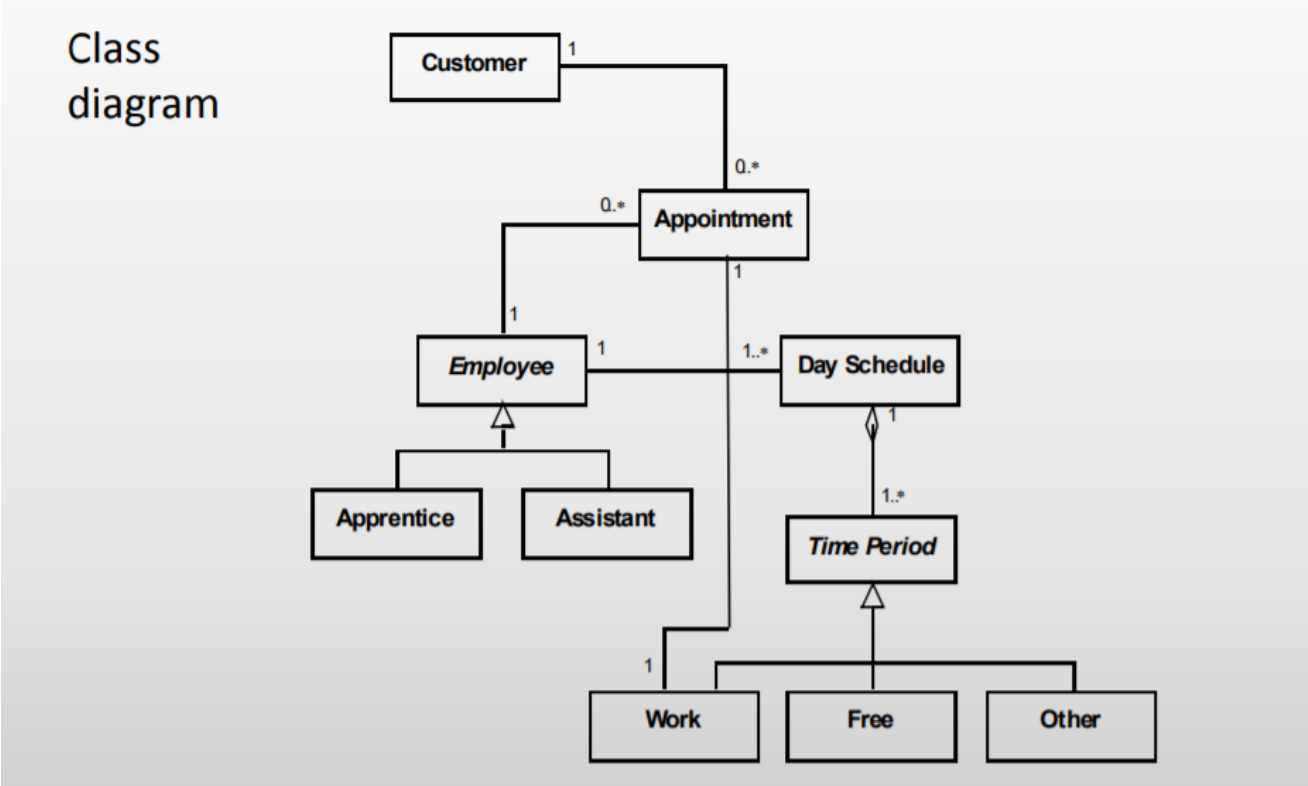
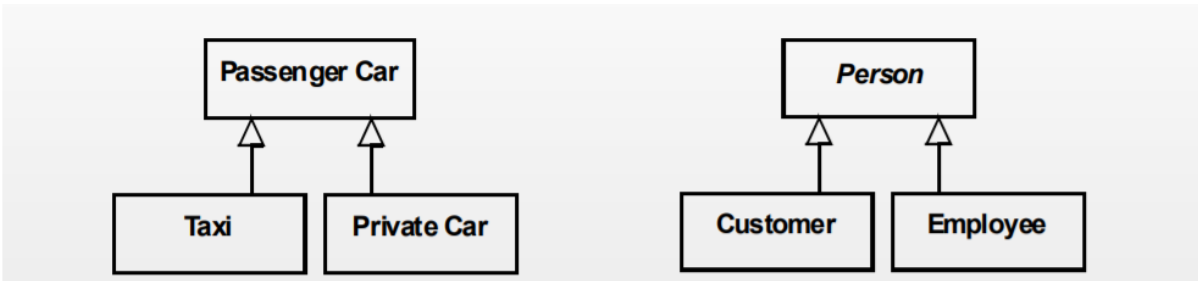
II: 4: Structure: Class diagrams and patterns, and how to use them. Patterns: Role, Relation, Hierarchy, Item-Descriptor.

Class Diagram



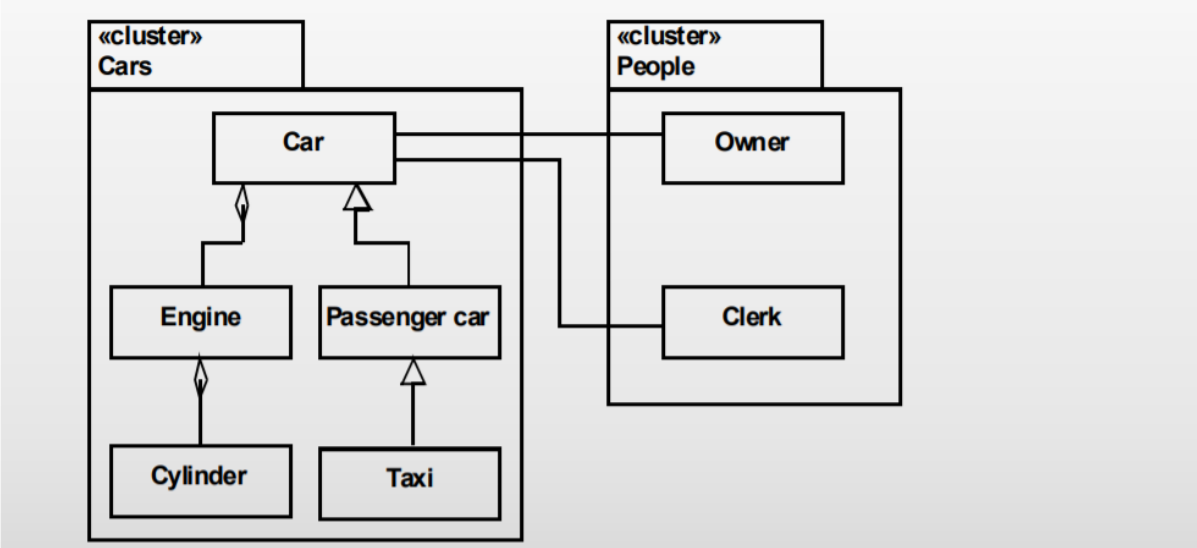
Generalization Structure



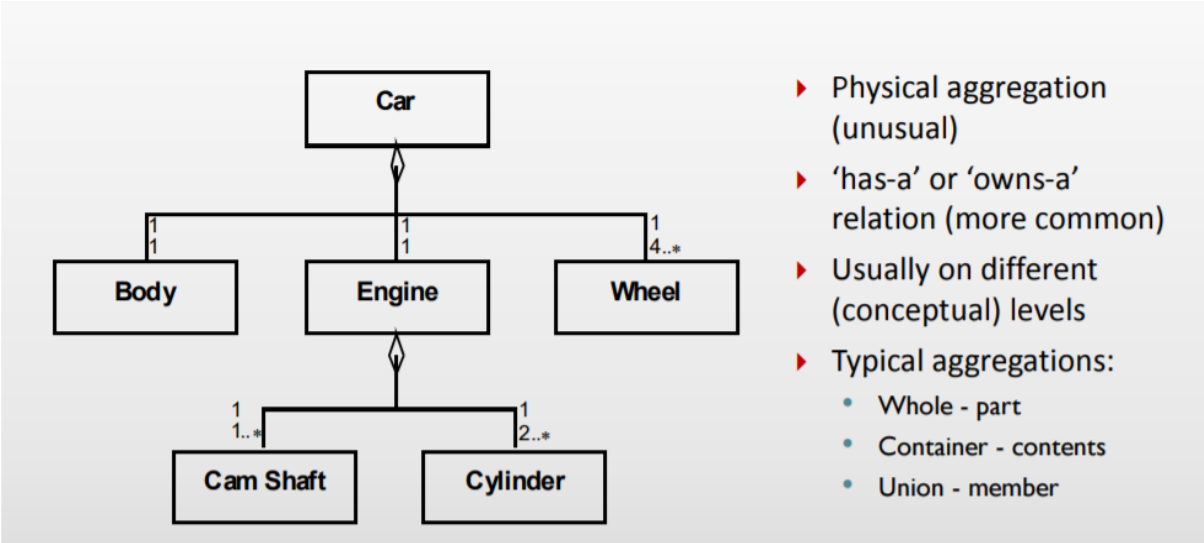
Is a taxi a passenger car? Is a private car a passenger car?

Generalization class. Specialization class. Abstract class (marked with italics, as seen on the above picture to the right).

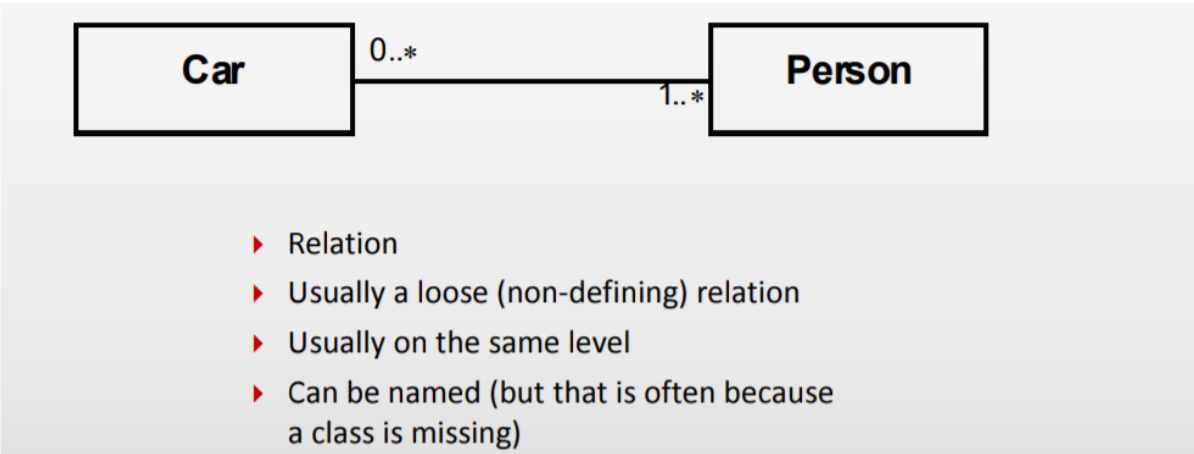
Cluster



Aggregation



The numbers show how many are involved. So for instance between wheel and car, it is 1 car per 4 wheels.



Car ownership relation: a person can have 0 to infinite cars.

Evaluate Systematically

Structures must be used correctly

- Generalization versus aggregation (is-a/has-a).
- Aggregation versus association (page 87).

Structures must be conceptually true

- Names, concepts, and structure reflect the user's understanding.
- The prospective user.

Structures must be simple

- Especially at the top levels.
- Avoid objects changing class. (customer turning into employee)
- Check against the system definition.

Generally

- Selecting the right structure is difficult
    - Try them out one by one for each pair of classes
    - Use the criteria to select the most correct structure
- It is very easy to include too many structures
    - Try to simulate functions and see if you can get to the relevant objects

Structure

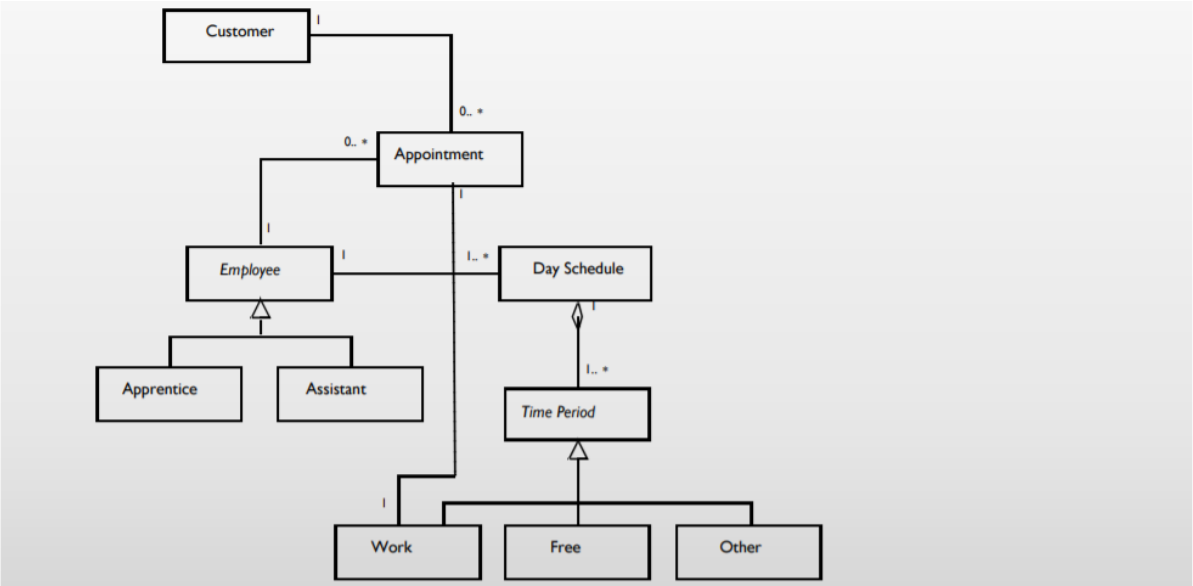
- Class structures
    - Generalization
    - Cluster
- Object structures
    - Aggregation
    - Association

Association = ----------

Aggregation = ------<>

Generalization = ---------->

Discuss the Diagram



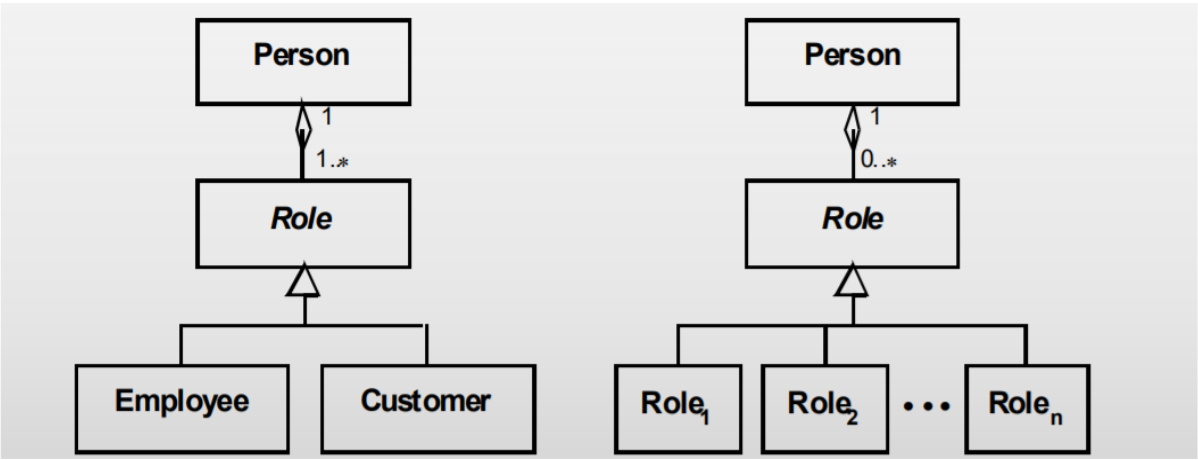
Customer and Appointment is an aggregation: they cannot stand without the other. Free is confusing. Apprentice can turn into assistant: not allowed.

Se mere på slide 20

Explore Patterns

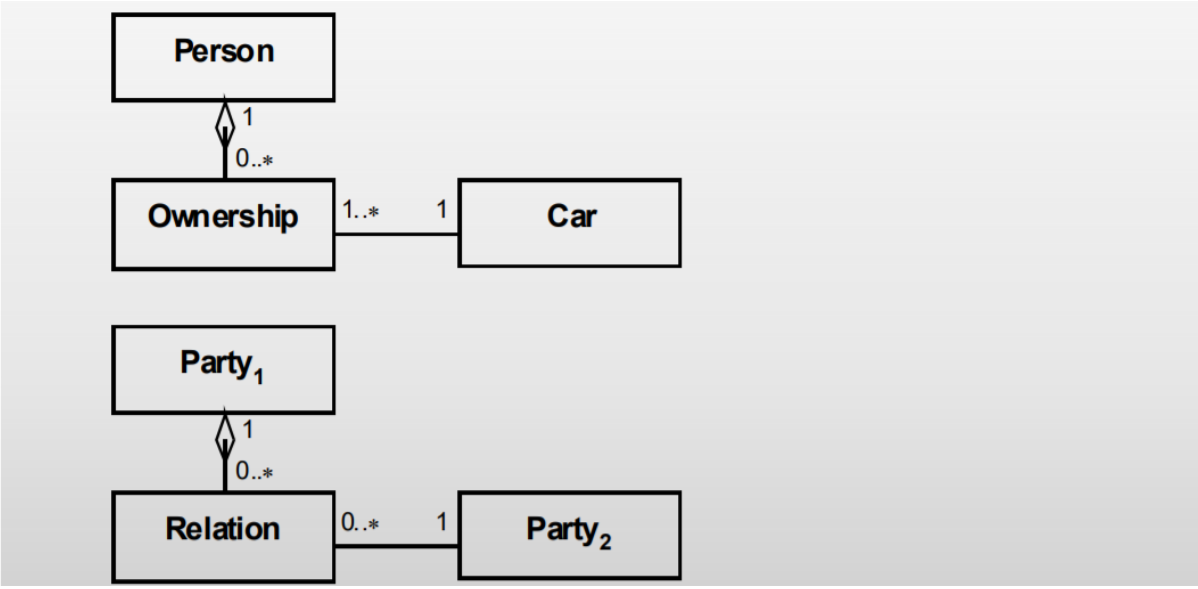
Role

- A person can have different roles
- The roles for a person change dynamically over time



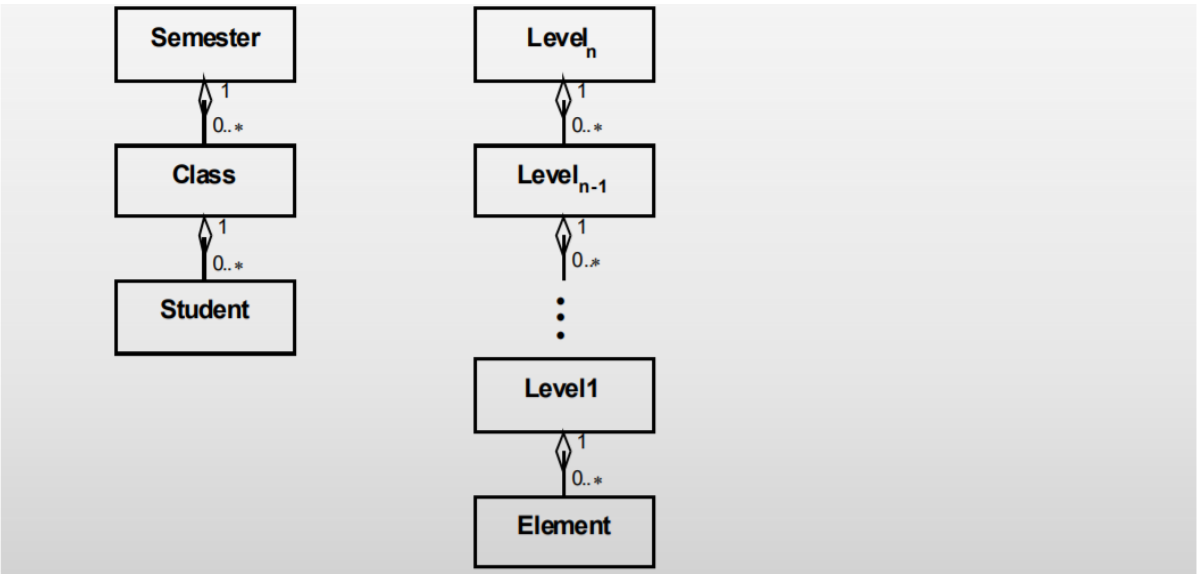
Relation

- Objects from two classes are related



Hierarchy

- Objects from different classes form a hierarchy



Item-Descriptor

- Properties of objects from one class (items) are described in an object from another class (descriptor). As in a library: book - is that the physical book or the one that is registered in the IT system.