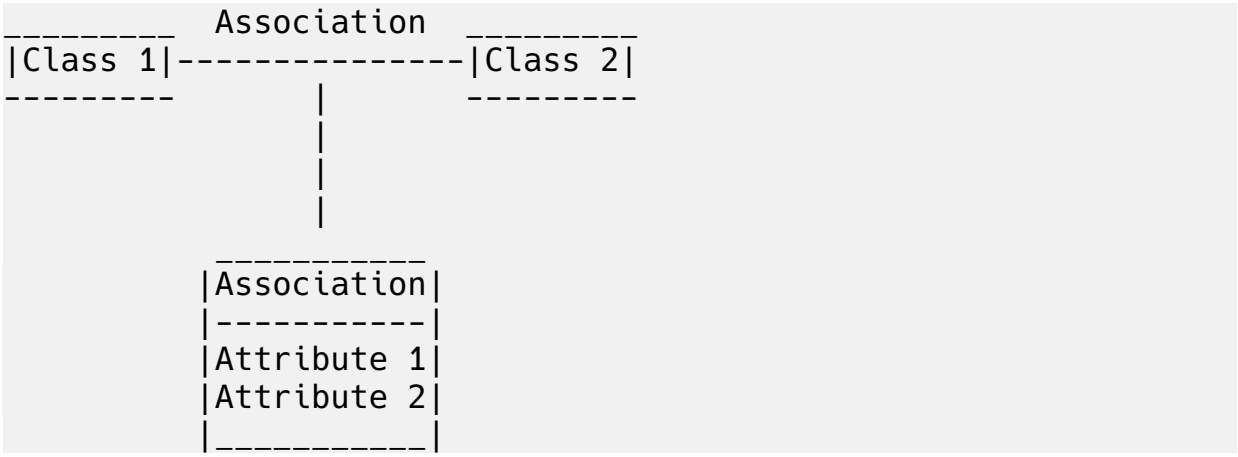
UML Data Modeling II

Association Classes

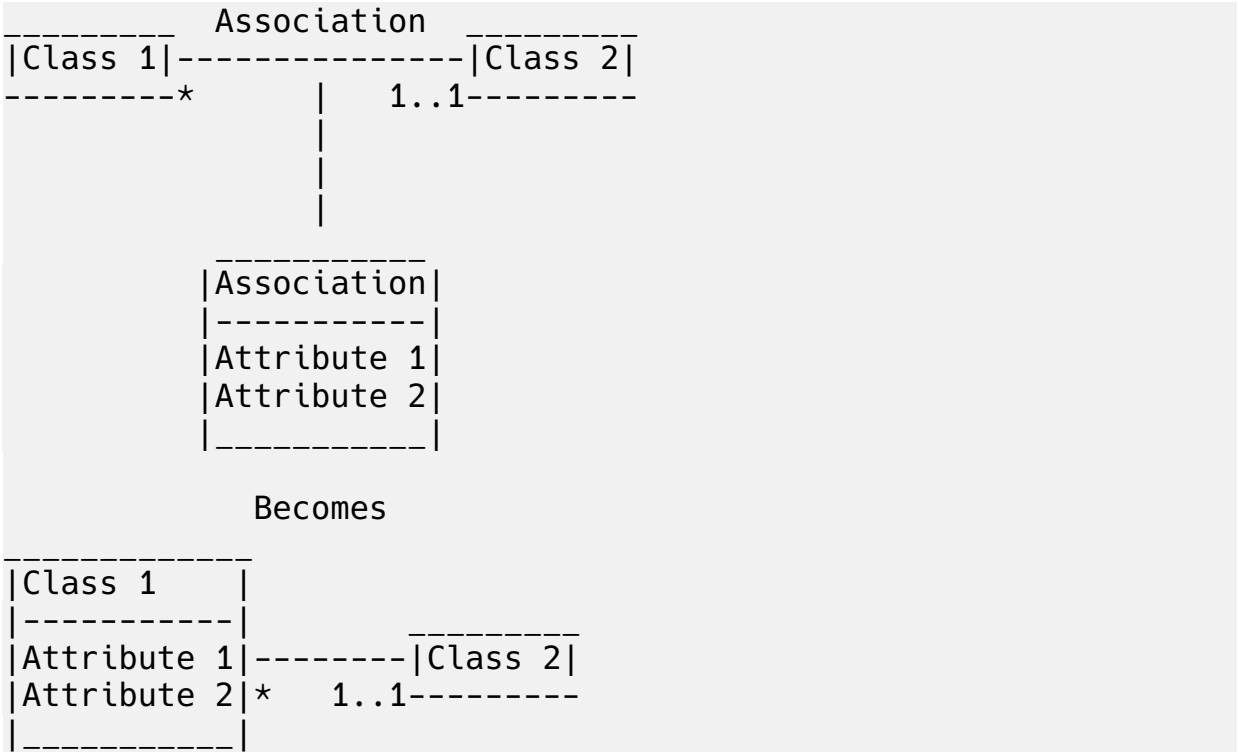
- Relationships between objects of two classes, *with attributes on relationships*



- The name may be placed in the association, in the class or in both
- It only captures **one** relationship between the two specific objects across the two classes

Eliminating Association Classes

- Unecessary if **0..1** or **1..1** multiplicity



Self-Associations

- Associations between a class and itself
    - A many-to-many relation between students representing siblings
    - A many-to-one relation between children and their mother (class person)

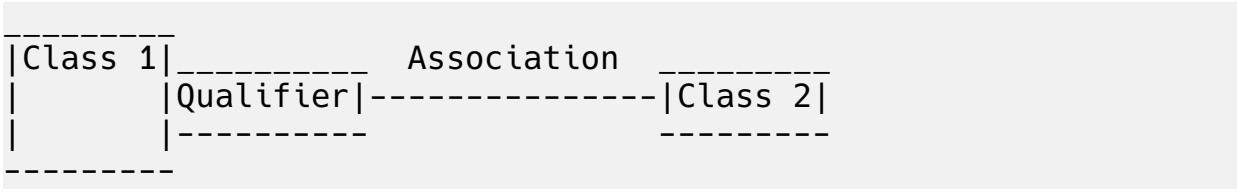
Association n-arys

- Associates n classes.
- Should be read in the form: for each of the other classes as a tuple, we can have at least i and at most j objects of the remaining single class.

Association vs attribute

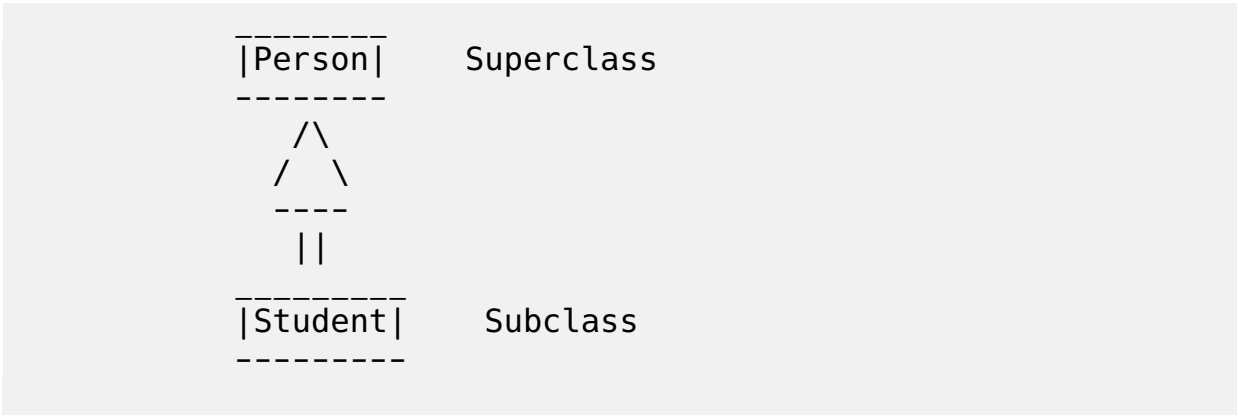
- An attribute should **never** be a reference to a class

Qualified associations



- Qualifier
    - One or more attributes of an association used to navigate from 1 to 2
    - "Access key" to 2 from an object of 1

Generalizations



- "is a " semantic relationship
    - A student "is a" person
- Going from a superclass to a subclass is a specialization
- Going from a subclass to a superclass is a generalization
- The subclass inherits the properties (attributes and relationships) of the

superclass and may add other

Generalization Properties

- Complete
    - If every object in the super class is in at least one subclass
- Incomplete or partial
    - If it's not complete
- Disjoint or Exclusive
    - If every object is on at most one subclass
- Overlapping
    - If it's not disjoing *Note: we can have any combination of the first two with the second two*

Aggregation

- Special type of association
- Models a "whole/part" relationship
- Represents a "has-a" relationship
- Does not link the lifetimes of a the whole and its parts
- 0..1 is implicit (on the whole part)
- Shared part can be included in several composites
- If some or all of the composites are deleted, shared part may still exist

Composition

- Strongest form of aggregation
- Strong ownership and coincident lifetime as part of the whole
- An object may be a part of only one composite at the time
- The whole is responsible for managing the creation and destruction of its parts
- 1..1 is implicit (on the whole part)

Constraints

- Specifies a conditions that has to be present in the system
- It is indicated by
    - an expression or text between brackets
    - note placed near (or connected by dotted lines to) the elements to which it relates

Derived Elements

- Element (class, attribute or association) computed using other elements in the model

- Notation: '/' before the name of the derived element
- Usually have an associated constraing that relates them with other elements