

Class diagram

- ✱ An example: how to draw a class diagram
- ✱ Operations
- ✱ Associations, association generalization
- ✱ Aggregation
- ✱ Composite objects
- ✱ Association classes

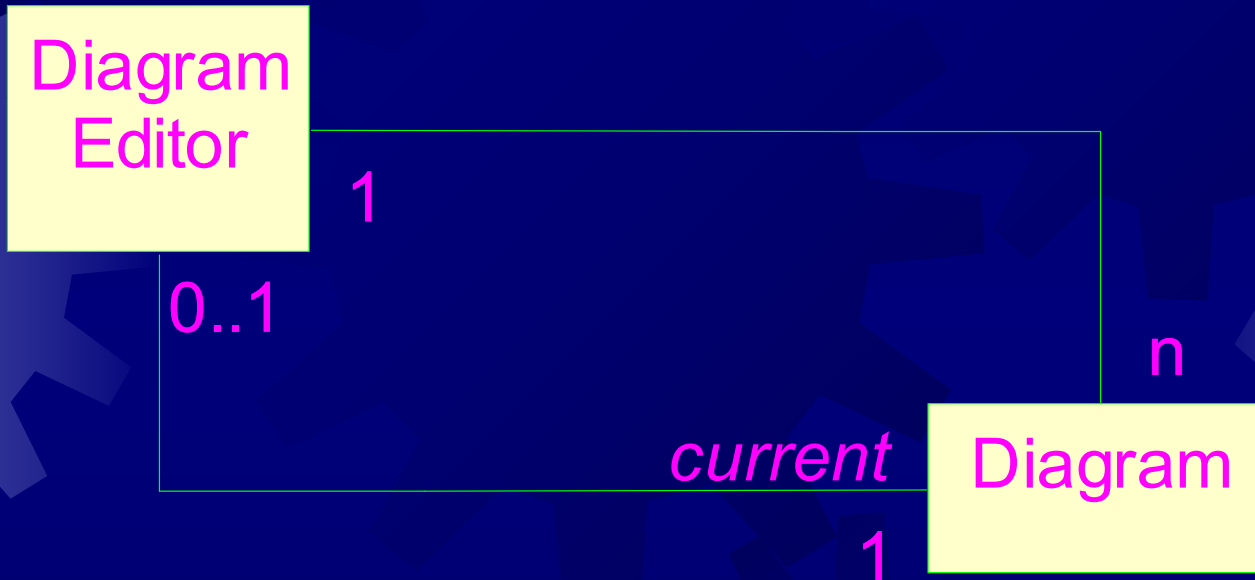
Classes and associations

- There is direct correspondence between class associations and object links.

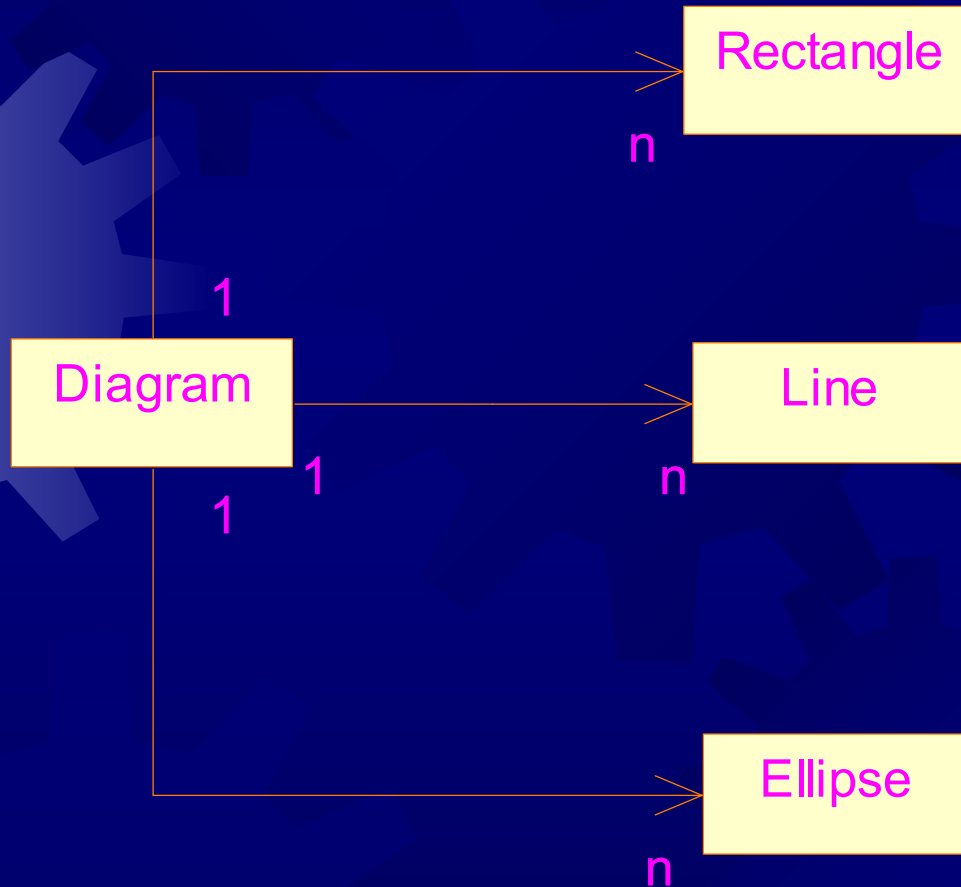
- Multiplicity: Demo



Multiple associations

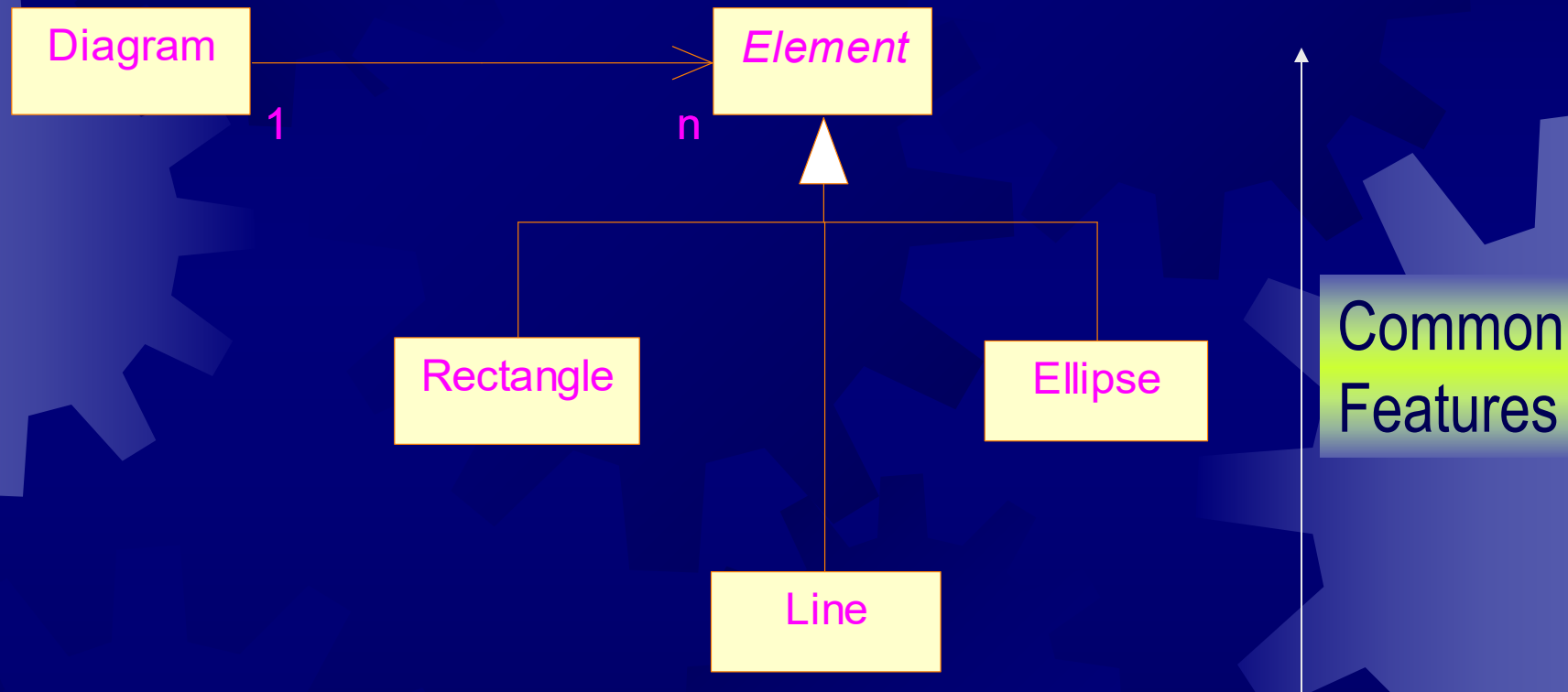


Class Generalization



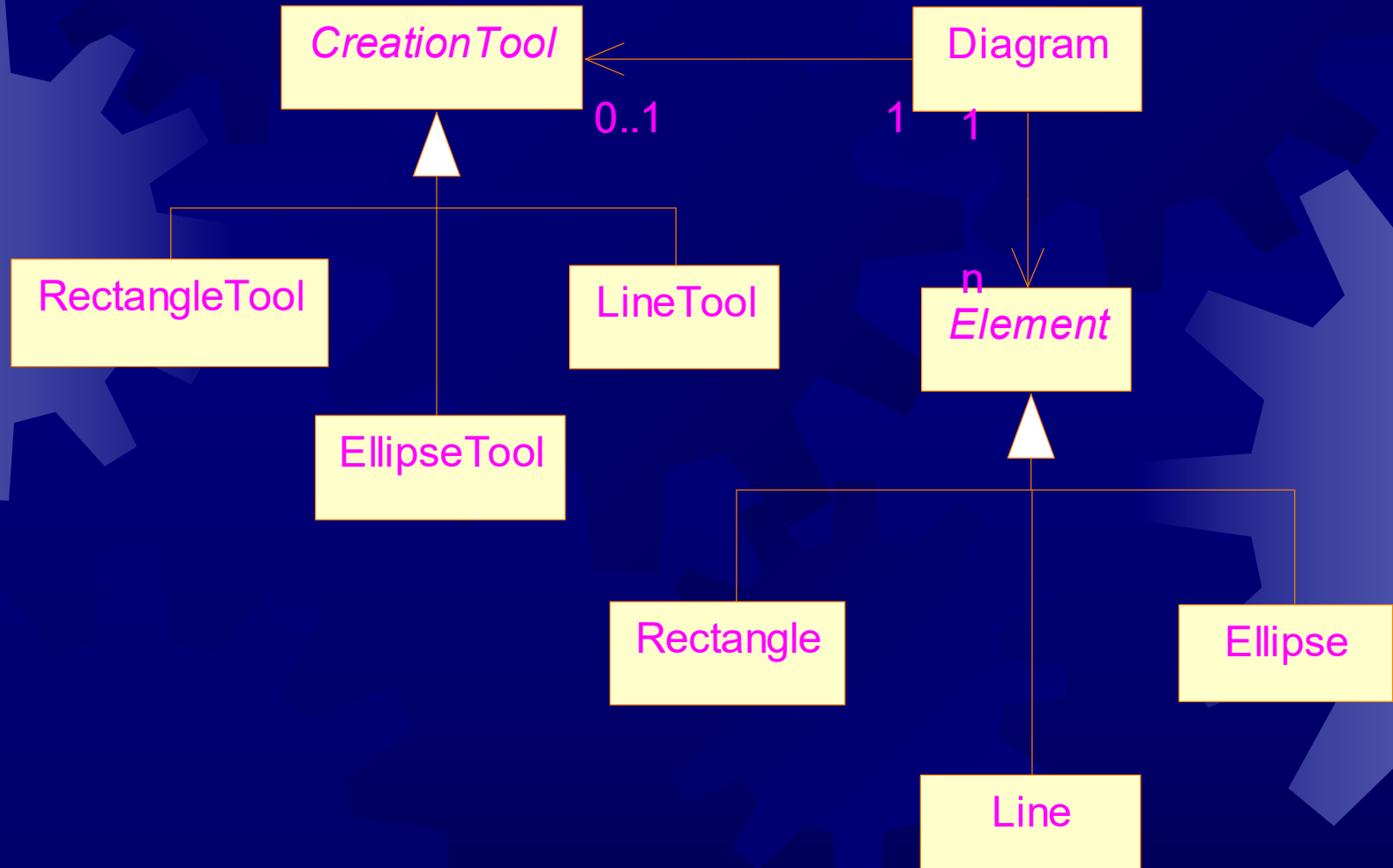
Direct implementation would lead to redundant Vectors

A more reasonable design



Does not introduce new relationship between objects

Similar for Creation Tools



The **instantiate** relationship





- ✱ Consider the relationship between RectangleTool and Rectangle
- ✱ The associate relationship is not applicable, since the connection is not permanent
- ✱ The instantiate relationship



Attributes and Operations

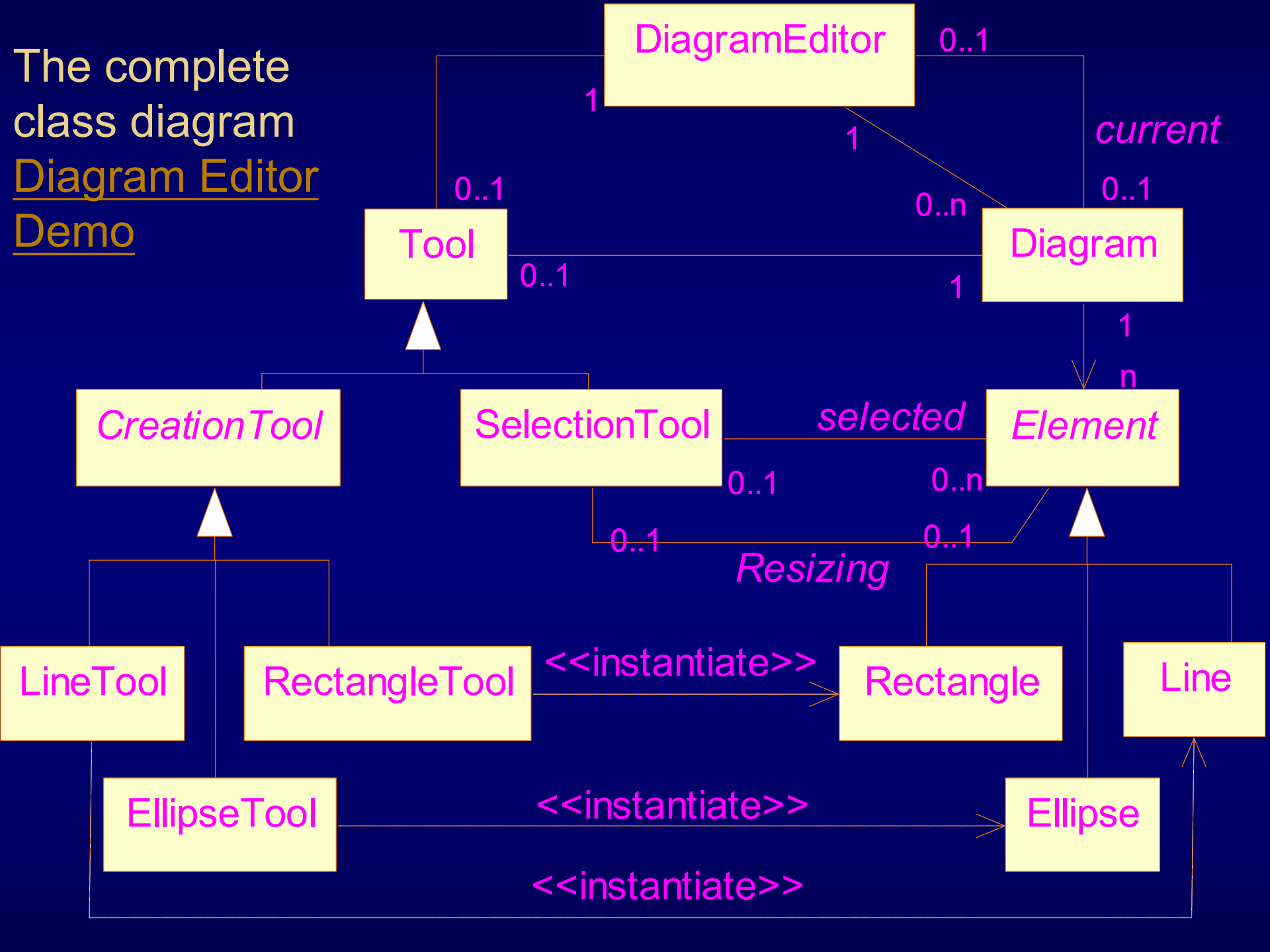
Element

  position[*] : Point

-  contains(Point) : Boolean
-  findControl(Point) : Boolean
-  move(Point)
-  moveControl(Point)

- Variable number of data elements.
- Format of operations.

The complete
class diagram
Diagram Editor
Demo



Multiplicity

- ★ Definition: the number of times a given entity can occur in some context.
- ★ Notation: number ranges
 - ★ E.g., 0..9
 - ★ * → infinity
 - ★ Abbreviation, e.g., 1..1 → 1
 - ★ Could be a list, e.g., 0, 3..*

Class multiplicity

- ★ Definition: the number of instances of the class that can exist.

The relationship of “instance of” is rarely shown.

- ★ Default: 0..*
- ★ Notation for singleton class

University 1

Attributes of classes

- ★ Attributes format

- ★ In the early stages: **name**

- ★ Common format: **name: type**

- ★ Complete format: **name: type = default_value**

- ★ Other classes should not be used as the attribute types → association







- ★ Instance scope

- ★ Class scope: underlined

- ★ Multiplicity of attributes

An example of class attributes

Module

-  mcount : Integer
-  code
-  title : String
-  credits : Integer = 15
-  exam[0..1] : Date
-  staff[*] : String

Operations of classes

☀ Format:

- ☀ name(p:parameter_type,...): return_type
- ☀ The types are optional

☀ Instance scope and class scope

Module

- ◆ Module(code, name)
- ◆ Count() : Integer
- ◆ GetTitle() : String
- ◆ SetTitle(t : String)

Object identity

- ✦ It is an intrinsic feature of objects
Even two objects have the same set of values for all attributes, they have two different identity
- ✦ It is different from the identifying attribute of a class

Student

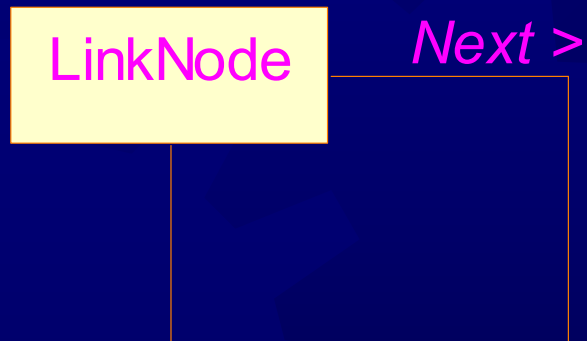
 name : String
 id : Integer

Associations

☀ Associations, role names, multiplicity

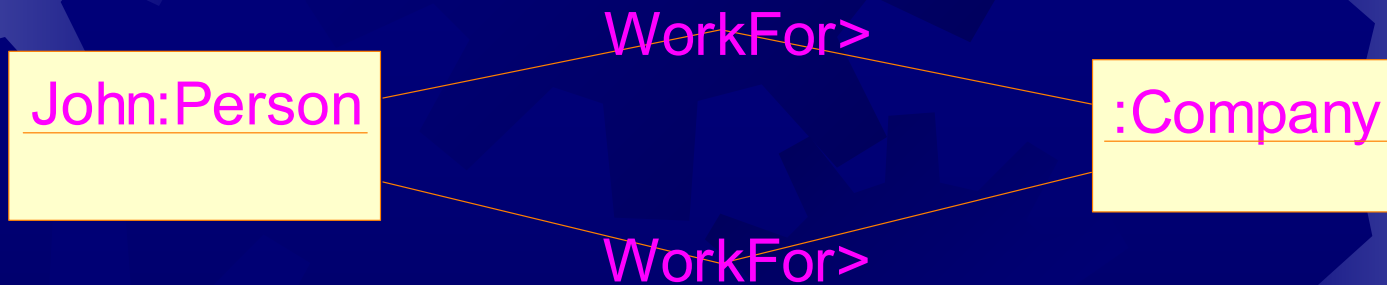


☀ Self-associations



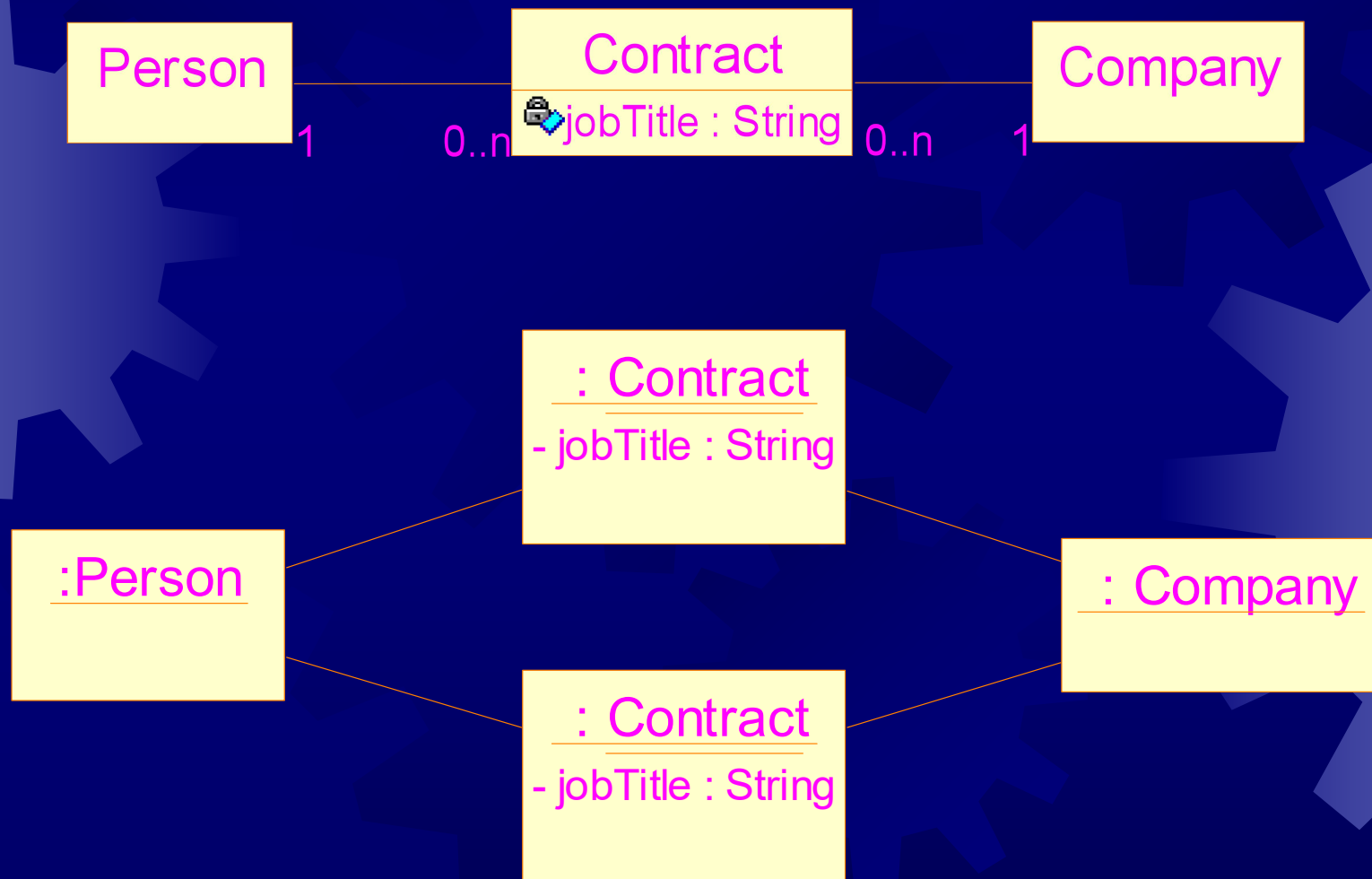
Links do not have identity

- ☀ The problem: John is employed as a lecture **and** a bar attendant by a company



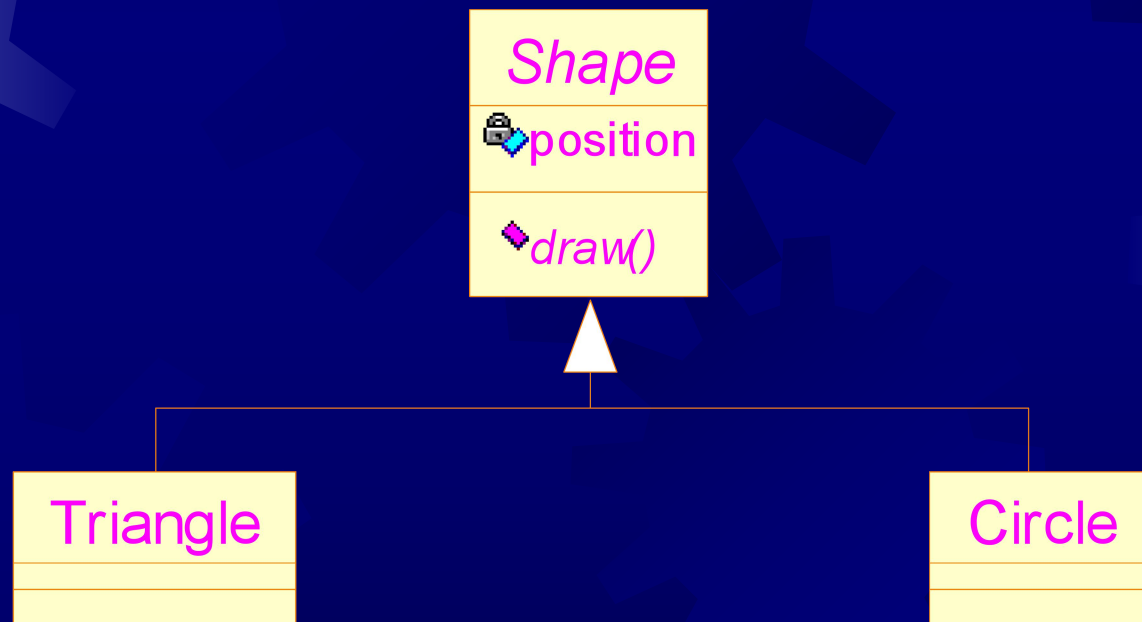
An illegal diagram in UML

Possible solution



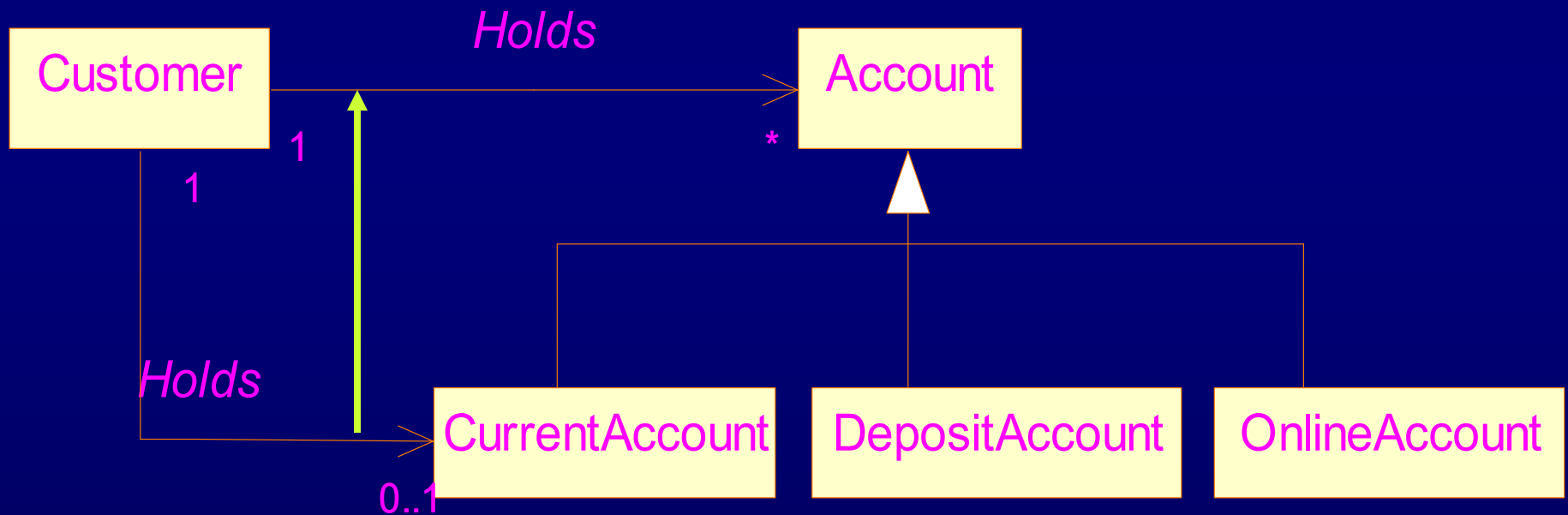
Generalization and specialization

- ✦ Super-classes (ancestors)
- ✦ Subclasses (descendants)
- ✦ Generalization/specialization
- ✦ Abstract classes



Association Generalization

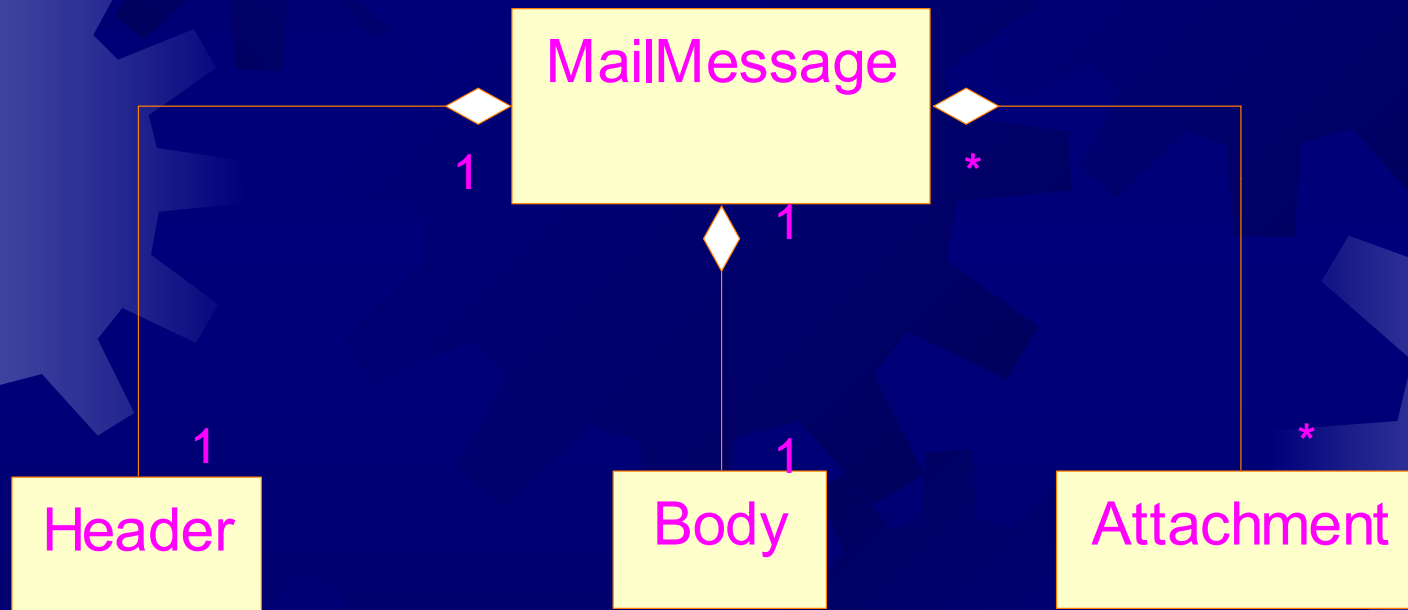
- ★ An example: a customer can hold at most one Current Account.



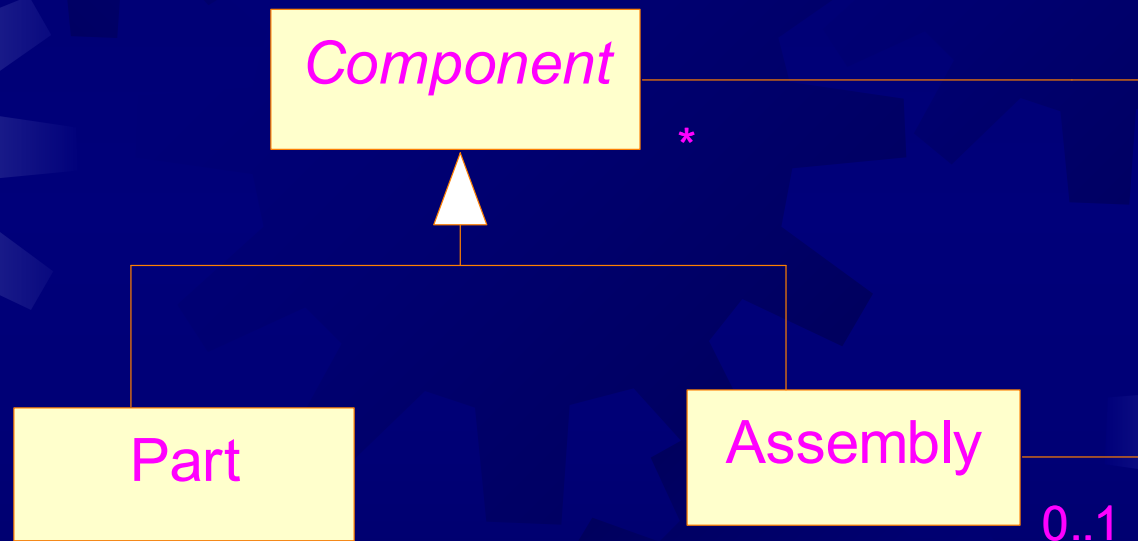
Aggregation

- ✱ Is just a kind of association
 - ✱ Most often it is used to describe part/whole relationship.
- ✱ Properties when objects are linked to instances of their own class.
 - ✱ Anti-symmetry: no self-connection
 - ✱ Transitivity: if A is a part of B, and B is a part of C, then A is also a part of C.

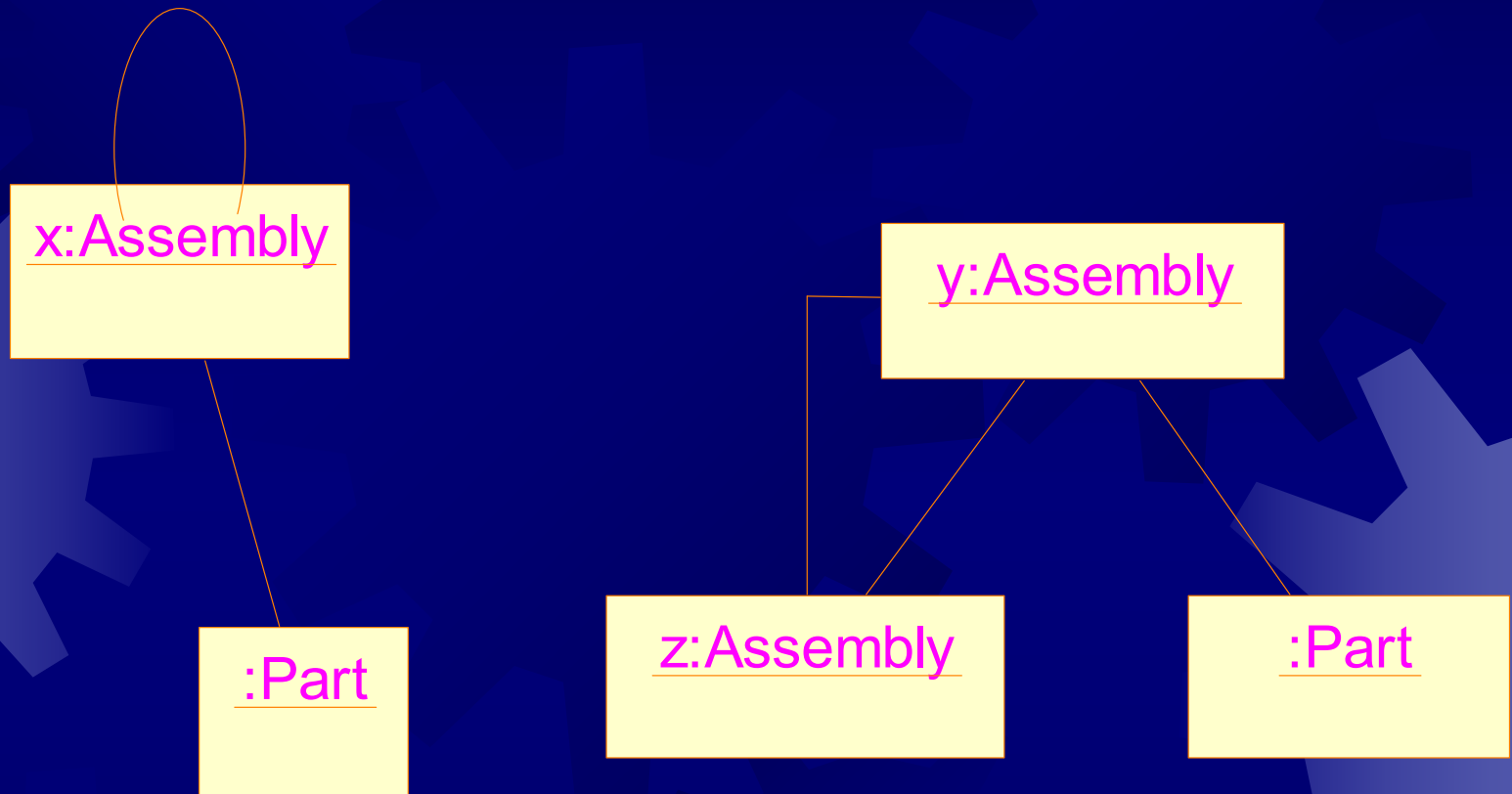
A “normal” aggregation relationship



If without aggregation

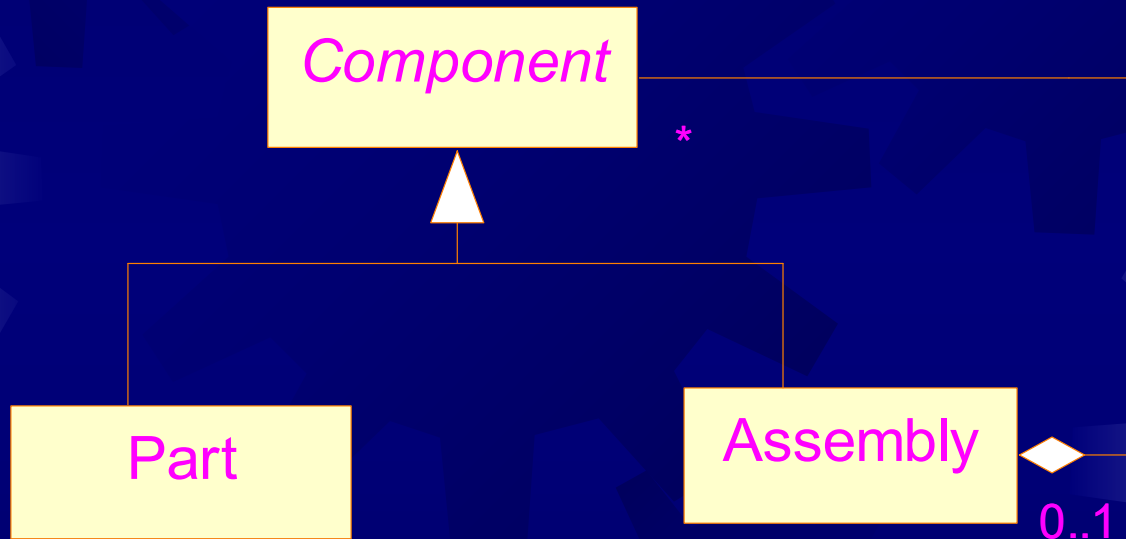


...leading to meaningless object diagrams



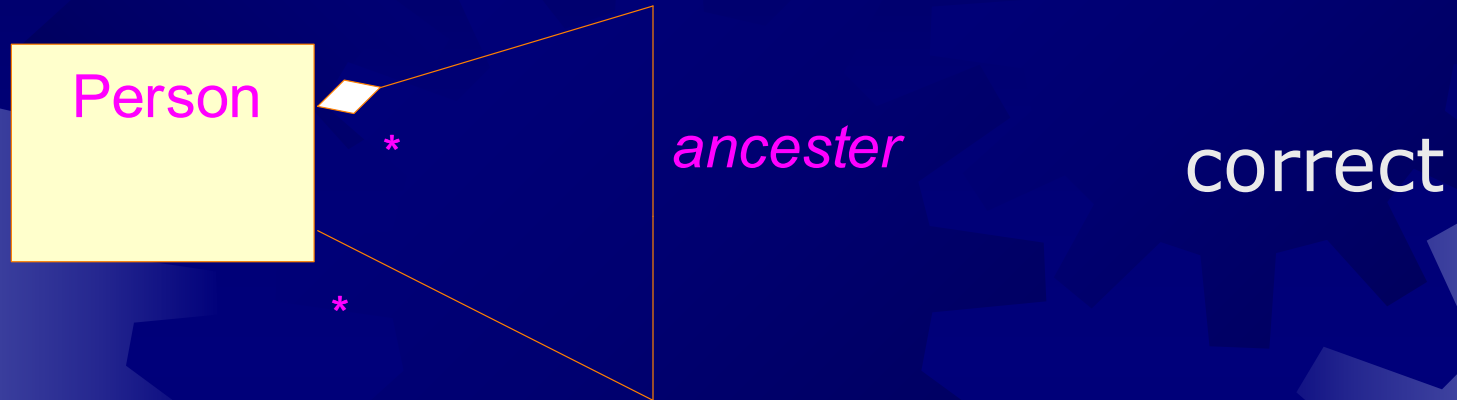
Cyclical object structures

Solve the problem with aggregation

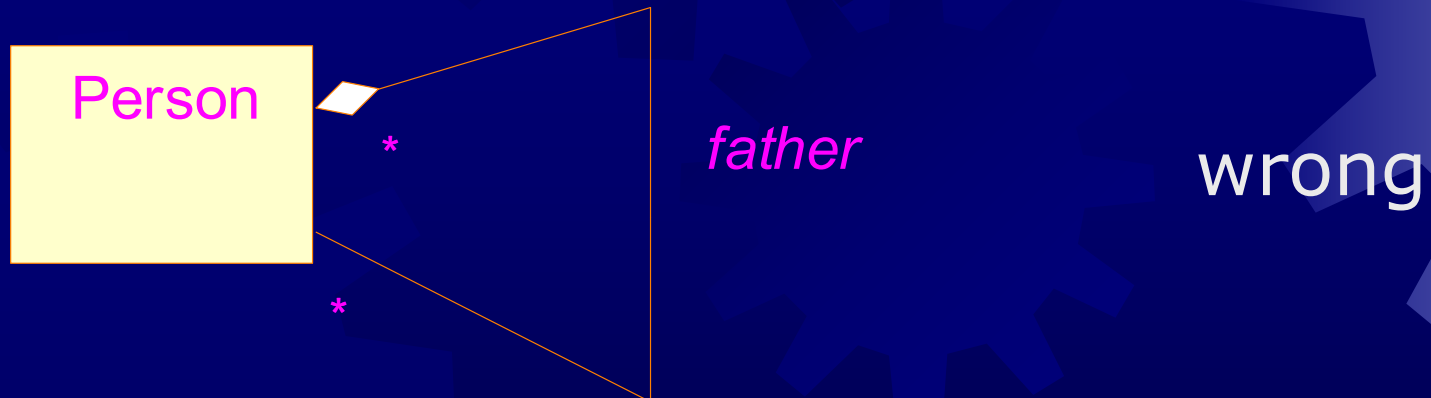


The two cyclical structures can be ruled out

Aggregation: not necessarily be part/whole relationship



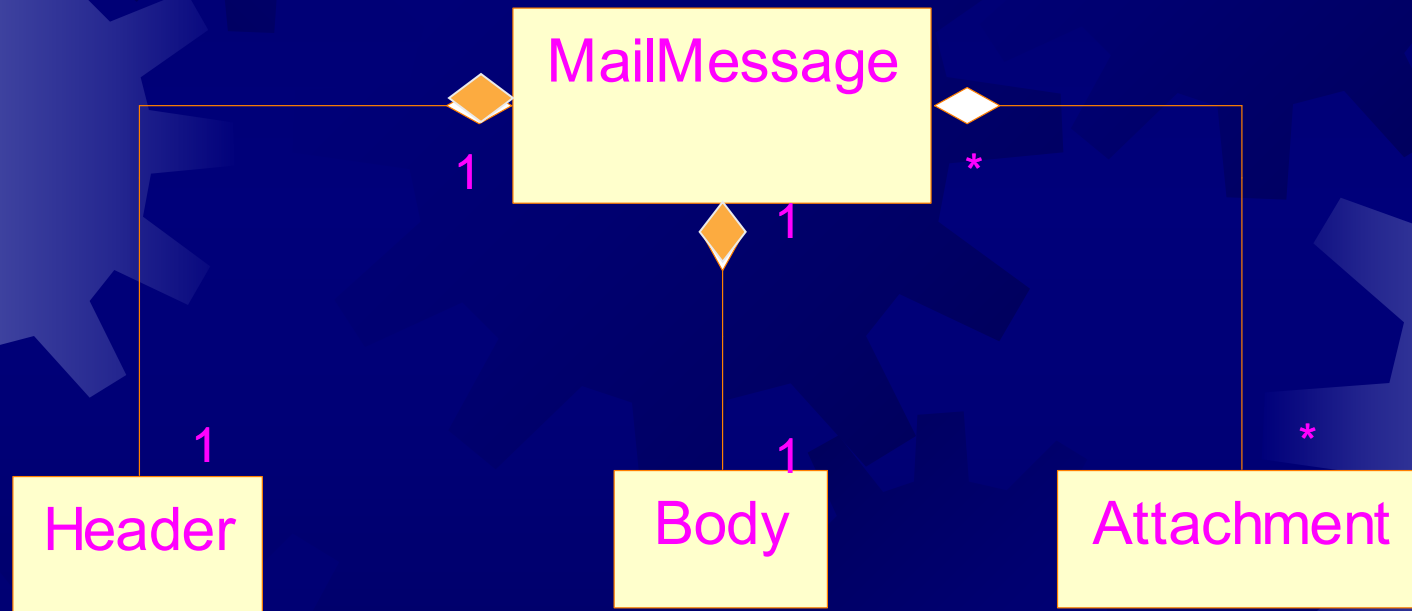
Two properties must be observed



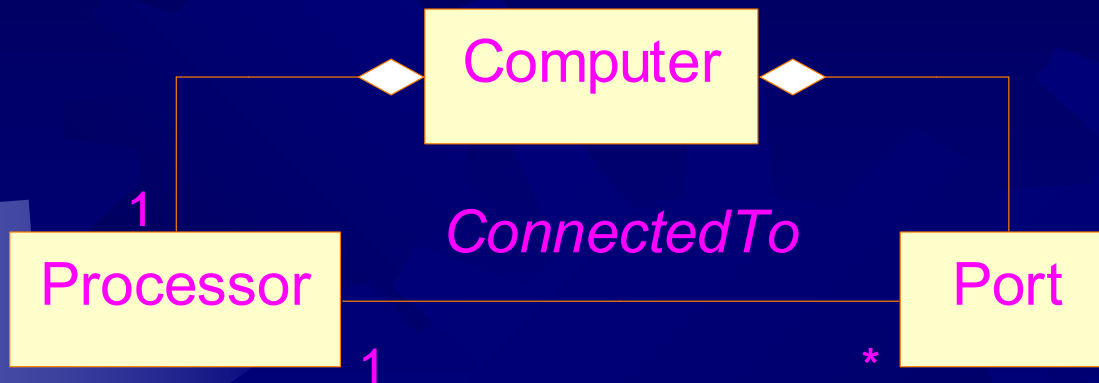
Composite objects

- ✱ Composite: a strong form of association
 - ✱ A “part” can only belong to one composite object
 - ✱ When a composite object is destroyed, all its dependent parts must be destroyed at the same time
 - ✱ UML notation: solid(black-color filled) diamonds

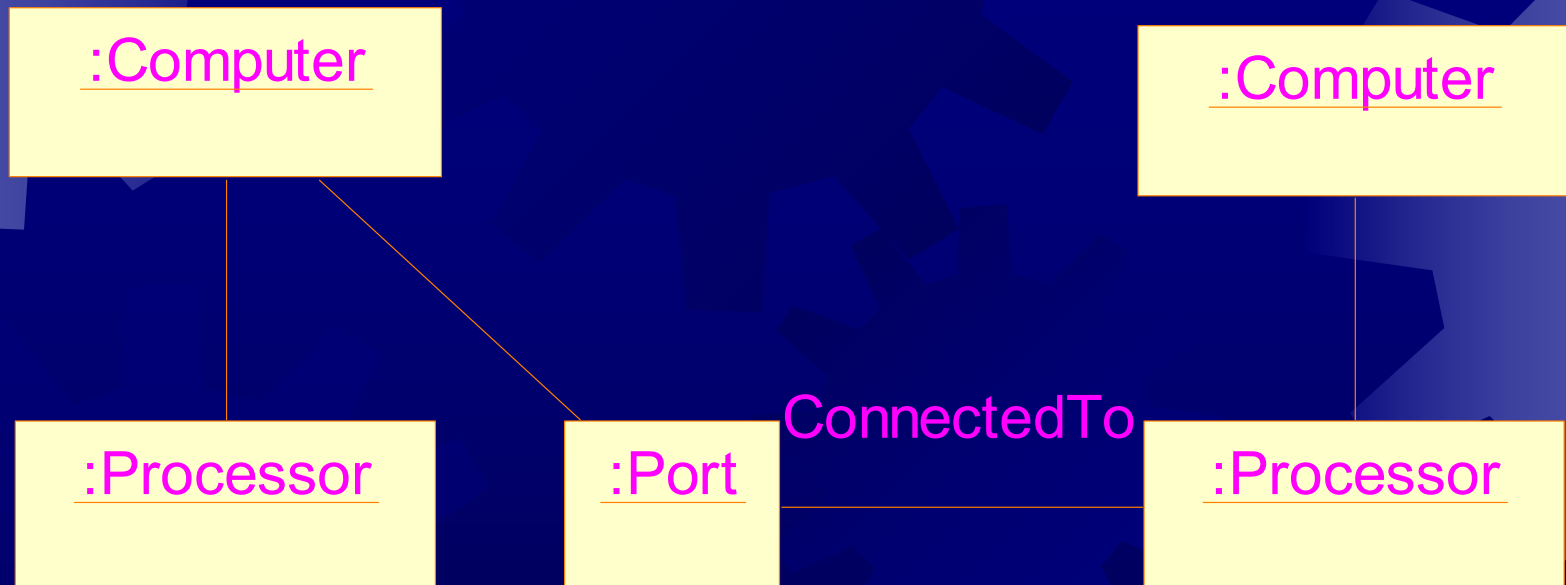
An example of composition



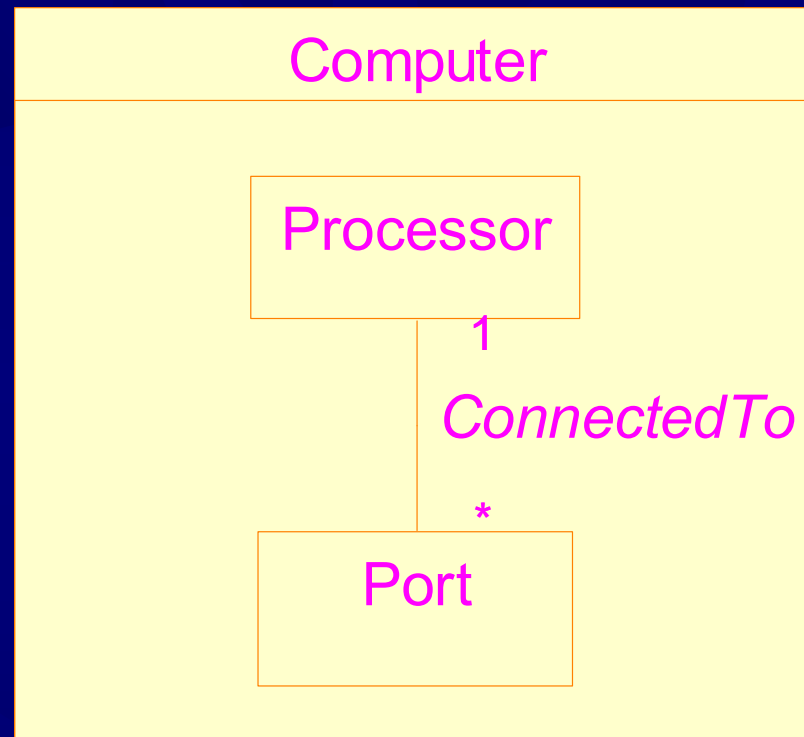
A situation where composition is not applicable



Structure of
a computer

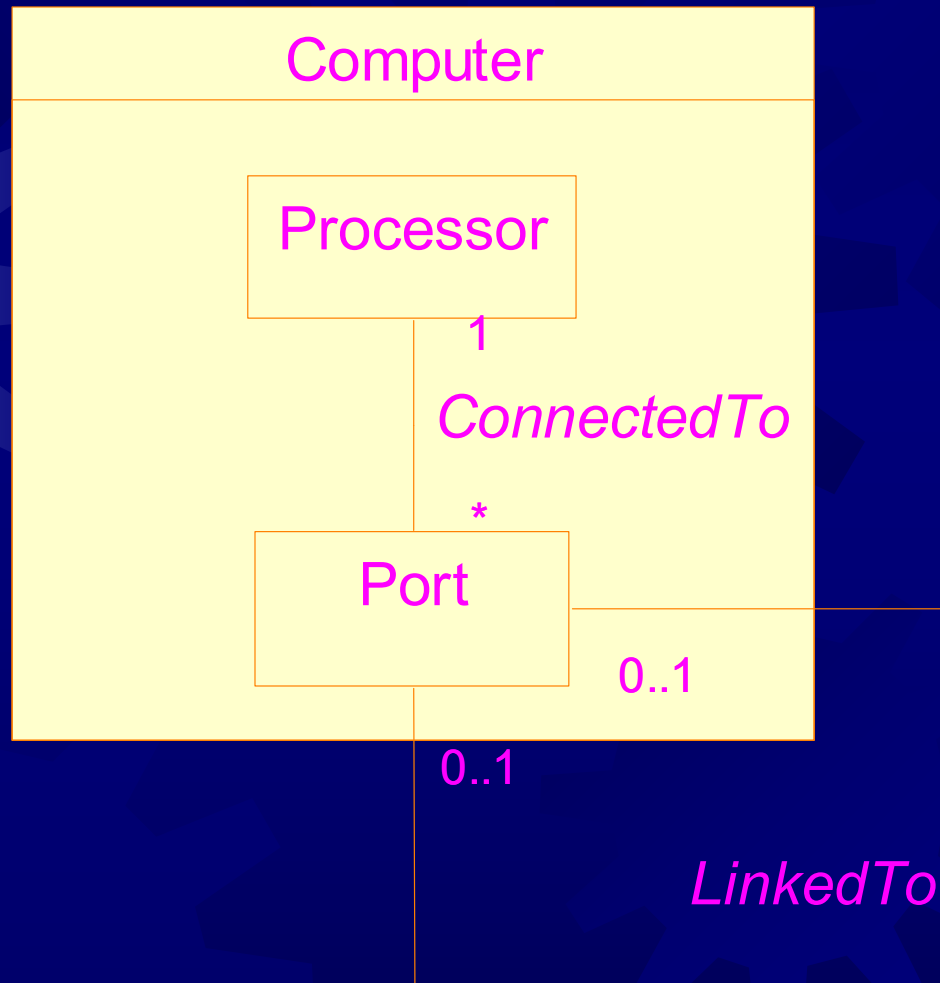


Another way to express the composition relationship



Associations inside a composite object can only link objects which are part of the same composite

To model a computer network



Association classes: the problem

- ✦ The problem: to record all the marks gained by students on all the modules that they are taking



- ✦ Drawbacks:
 - ✦ The marks attribute is not intrinsic for Student

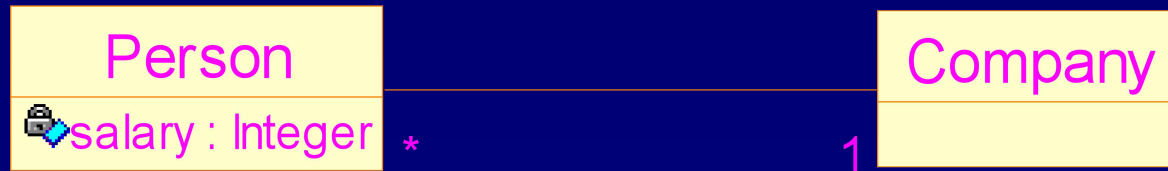
Association classes

- ✱ Associating data values with links
- ✱ UML notation: a dashed line to a link
- ✱ Applicability: to model attributes which are not intrinsic to the linked classes, but intrinsic to the link itself.

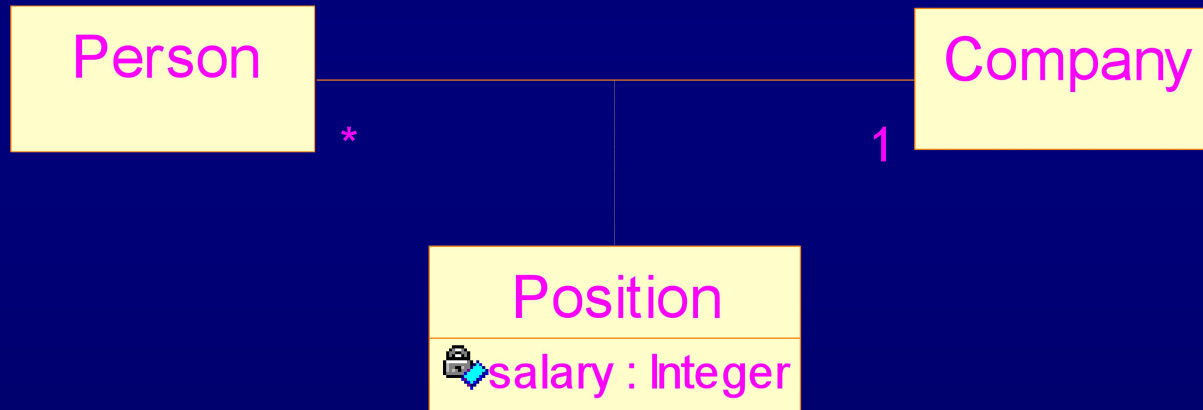
UML notation for association classes



Another example of using association class



Not preferred



More accurate

An association class can have associations with other classes

