



Case study (OOP in Python)



- UML: Unified Modeling Language
- Case Study

<http://danse.us/trac/engdiffracton/browser/t>

"UML" - Unified Modeling Language

* Relationship Lines Class Diagram



• Generalization

- Inheritance :  sub → super
- Implementation :  class → interface

3 major Relationship

"is a"

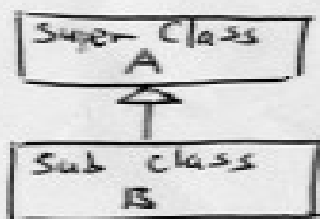
• Association : →

- Composition : 
- Aggregation : 

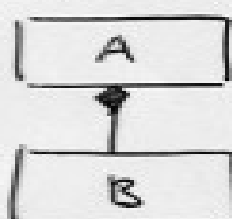
"has a"

• Dependency :

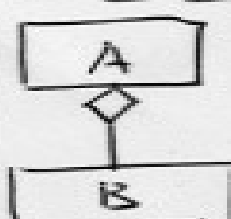
"use a"



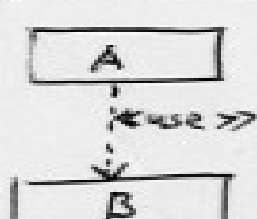
- Inheritance
- B is derived from A (B is A)



- Composition
- A contains B type object (A has B)



- Aggregation
- A aggregates B (A has B)



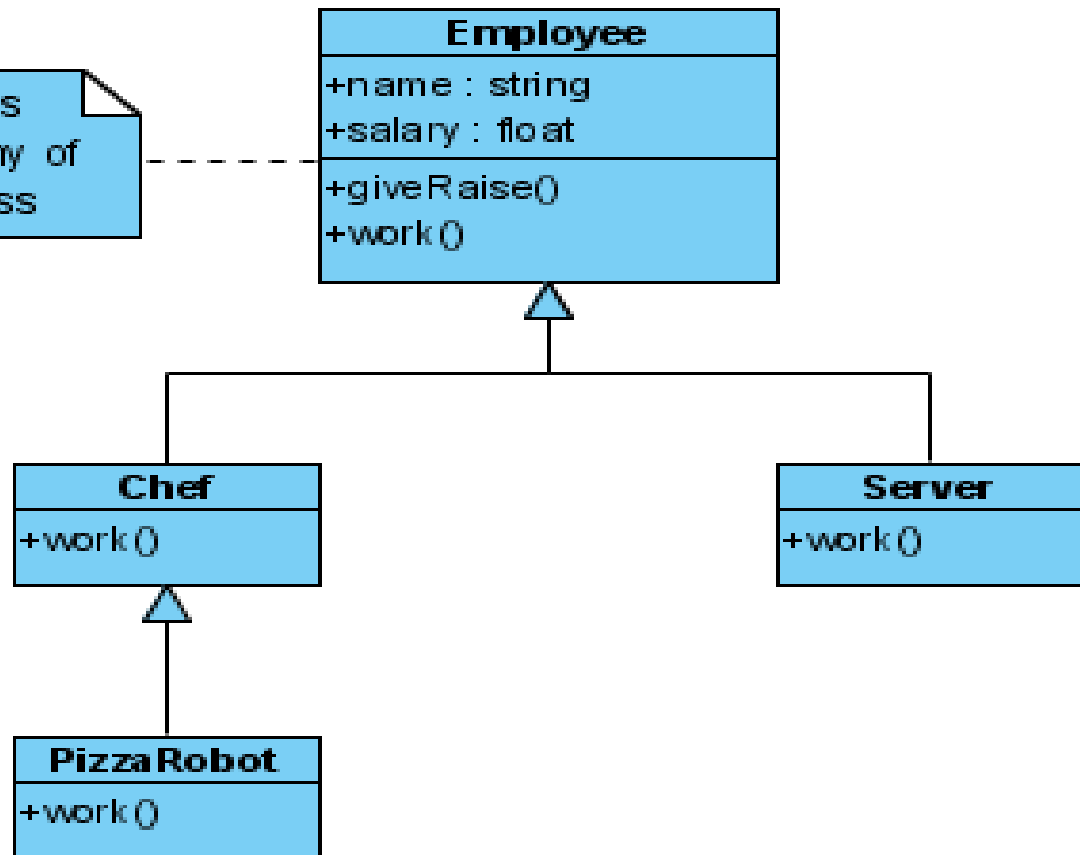
- Dependency
- A depends upon B (A uses B)

* Difference between composition & aggregation → Life cycle

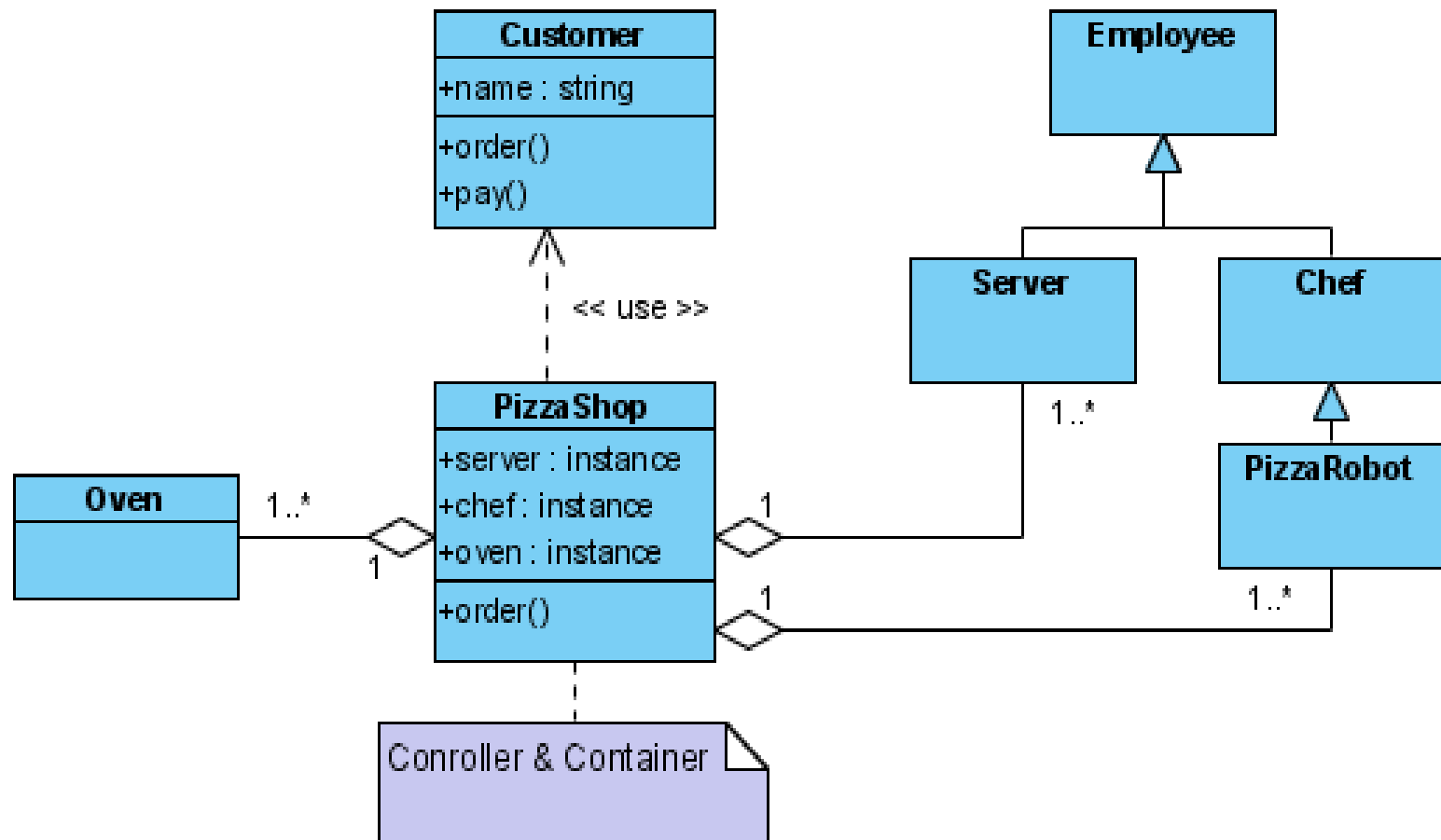
- Composition has stronger ownership than aggregation
- if A is destroyed, B is destroyed as well in composition
- Even if " " , B survives in aggregation

Employee

This example shows inheritance hierarchy of Employee base class



Pizzashop



Customer

