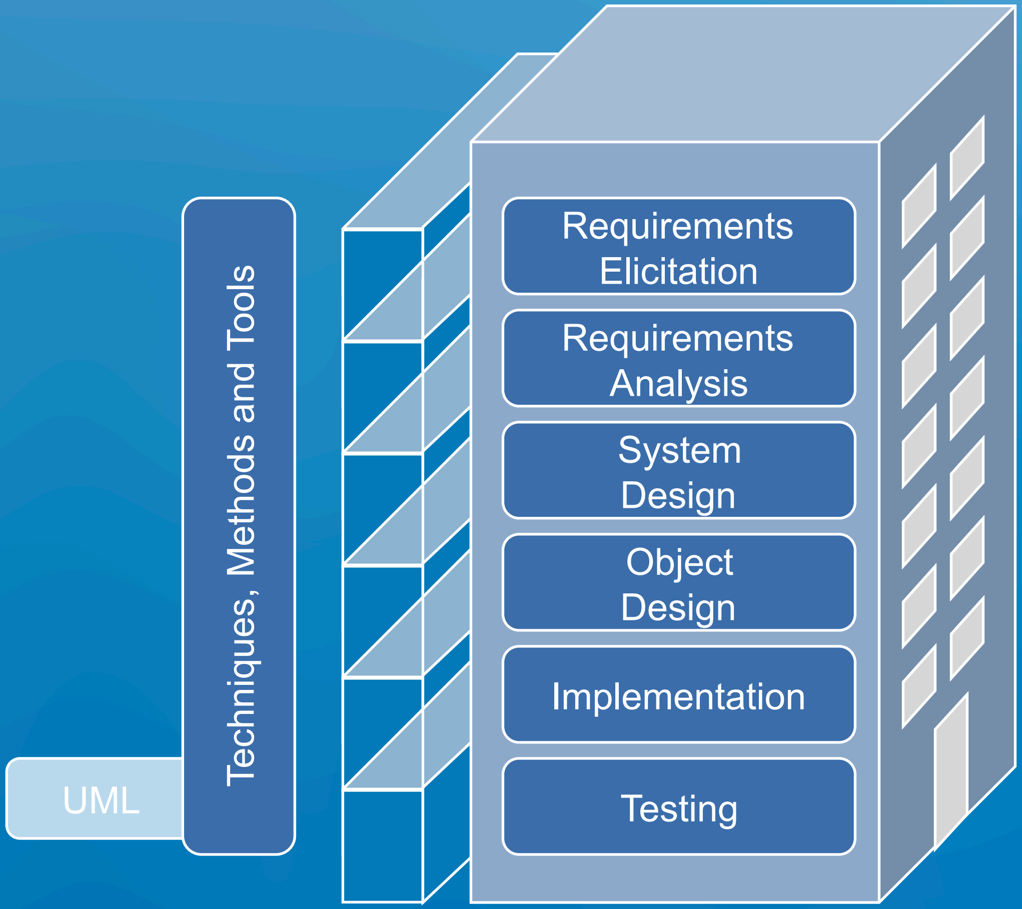
Software Engineering for Engineers

Lecture 1: UML Class Diagrams

Outline

- What is UML and why do we use it?
- UML Class Diagram
  - Associations
  - Inheritance
  - UML to Java

Where are we?


Requirements Elicitation
Requirements Analysis
System Design
Object Design
Implementation
Testing
Techniques, Methods and Tools
UML

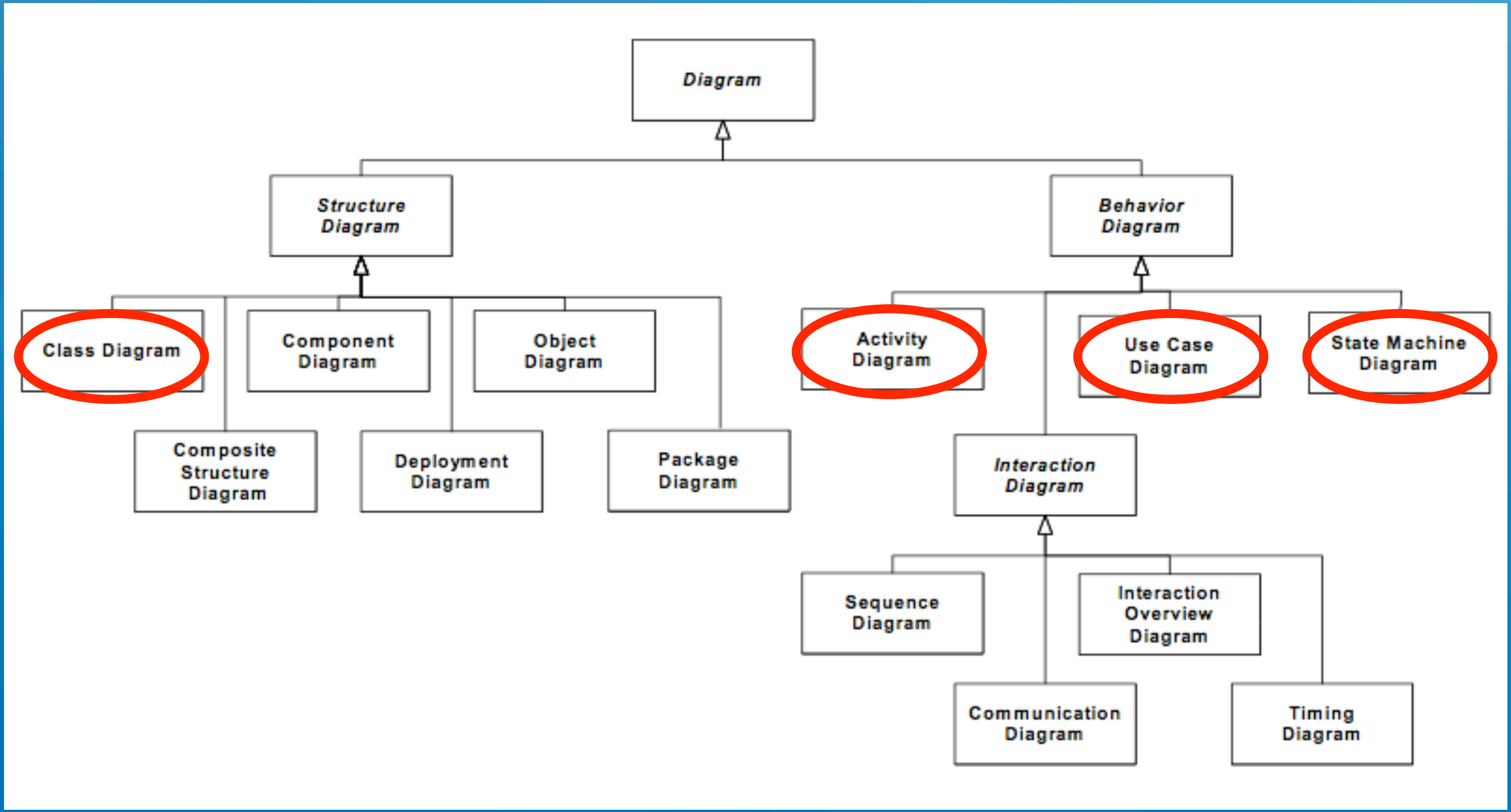
What is UML?

- UML (Unified Modeling Language)
  - Convergence of notations used in object-oriented methods
    - OMT (James Rumbaugh and colleagues)
    - Booch (Grady Booch)
    - OOSE (Ivar Jacobson)
- Current version 2.1.2
  - Information at the UML portal http://www.uml.org/
- Commercial CASE tools: Rational Rose (IBM), Together (Borland), Visual Architect (business processes, BCD)
- Open Source CASE tools: ArgoUML, StarUML, Umbrello, Unicase
- Commercial as well as Open Source: PoseidonUML (Gentleware)

We use Models to describe Software Systems

- **System model:** Object model + functional model + dynamic model
- **Object model:** What is the structure of the system?
  - UML Notation: Class diagrams
- **Functional model:** What are the functions of the system?
  - UML Notation: Use case diagrams
- **Dynamic model:** How does the system react to external events?
  - UML Notation: Sequence, State chart and Activity diagrams

Another view on UML Diagrams

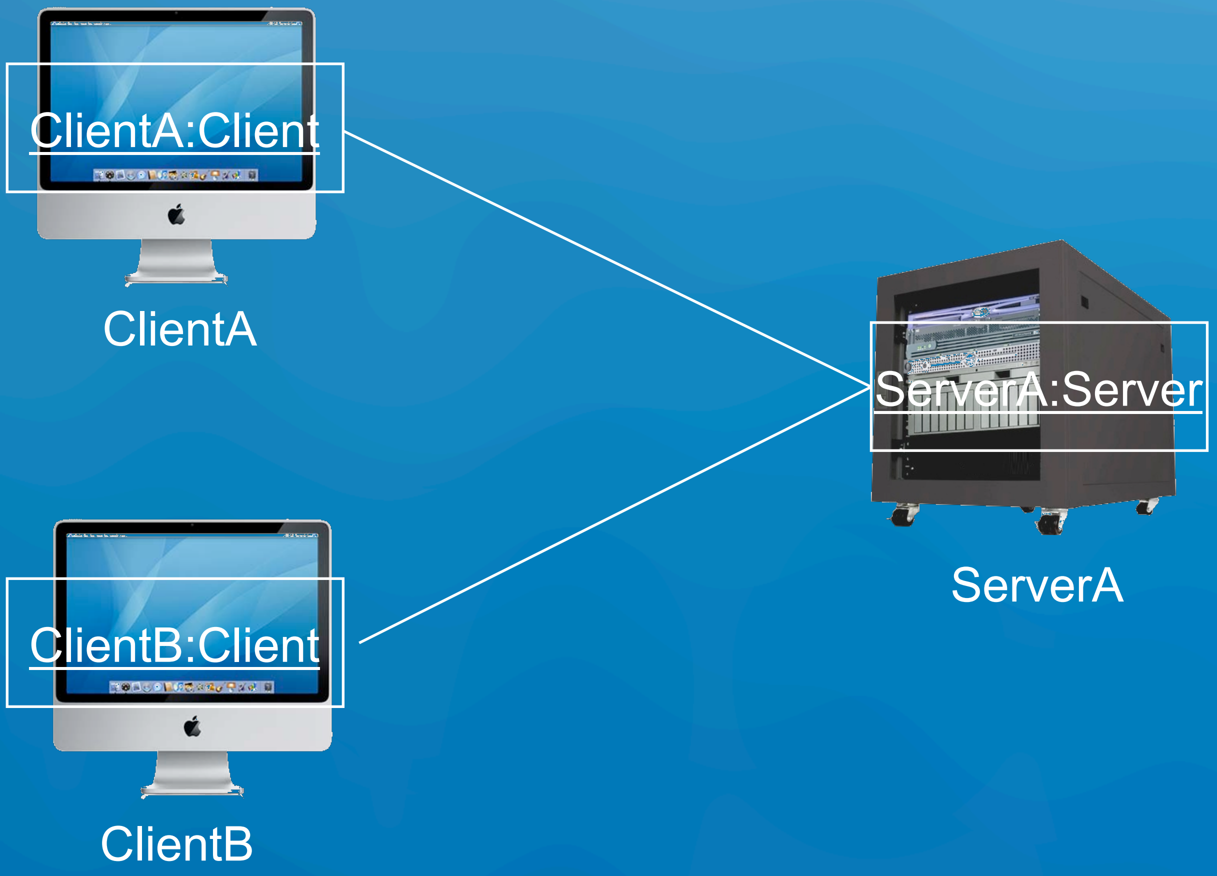


Where are we now?

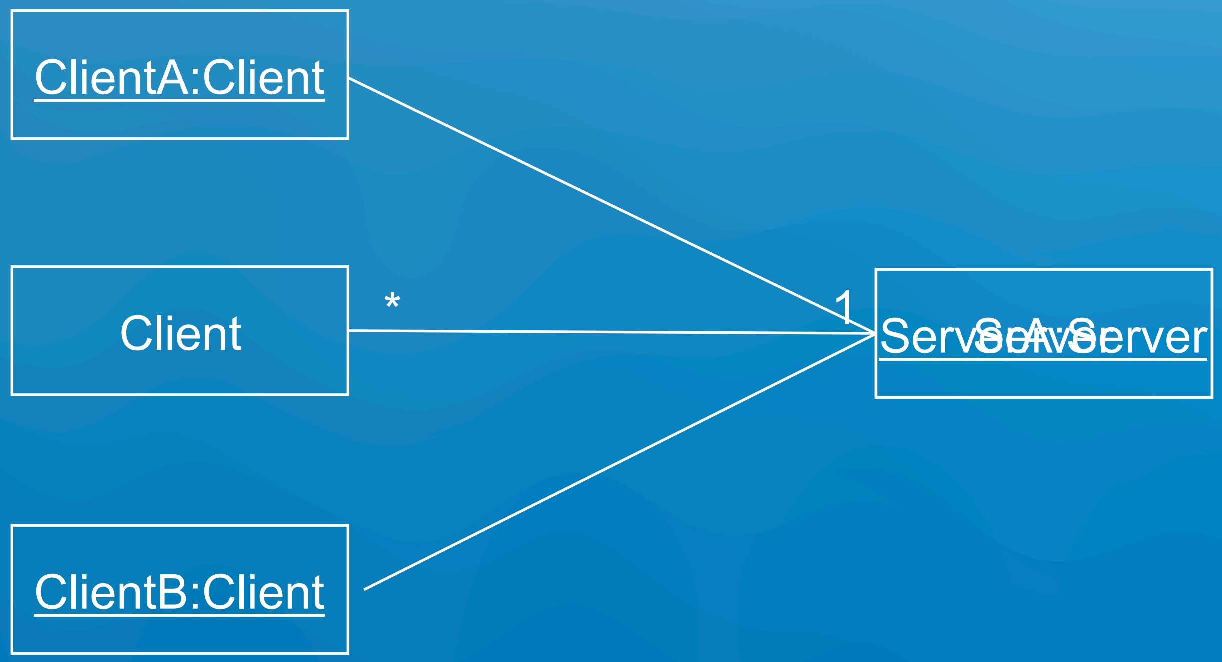
✓ What is UML and why do we use it?

- UML Class Diagram
  - Associations
  - Inheritance
  - UML to Java

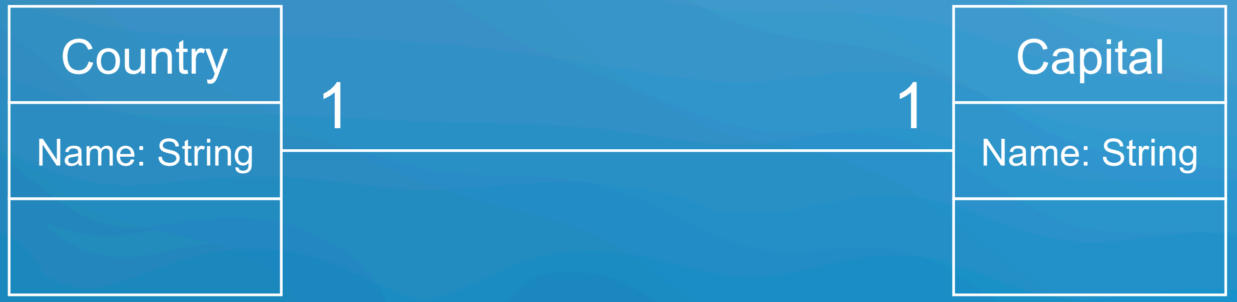
From an image to an Object Diagram



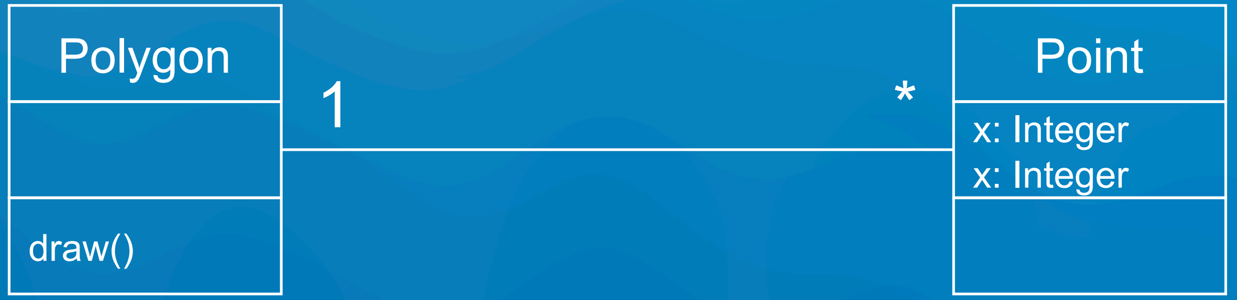
From an Object Diagram to a Class Diagram



1-to-1 and 1-to-many Associations

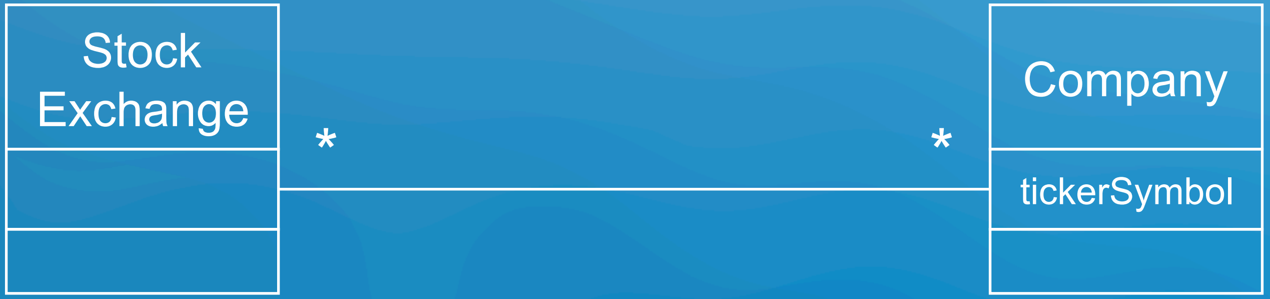


1-to-1 association



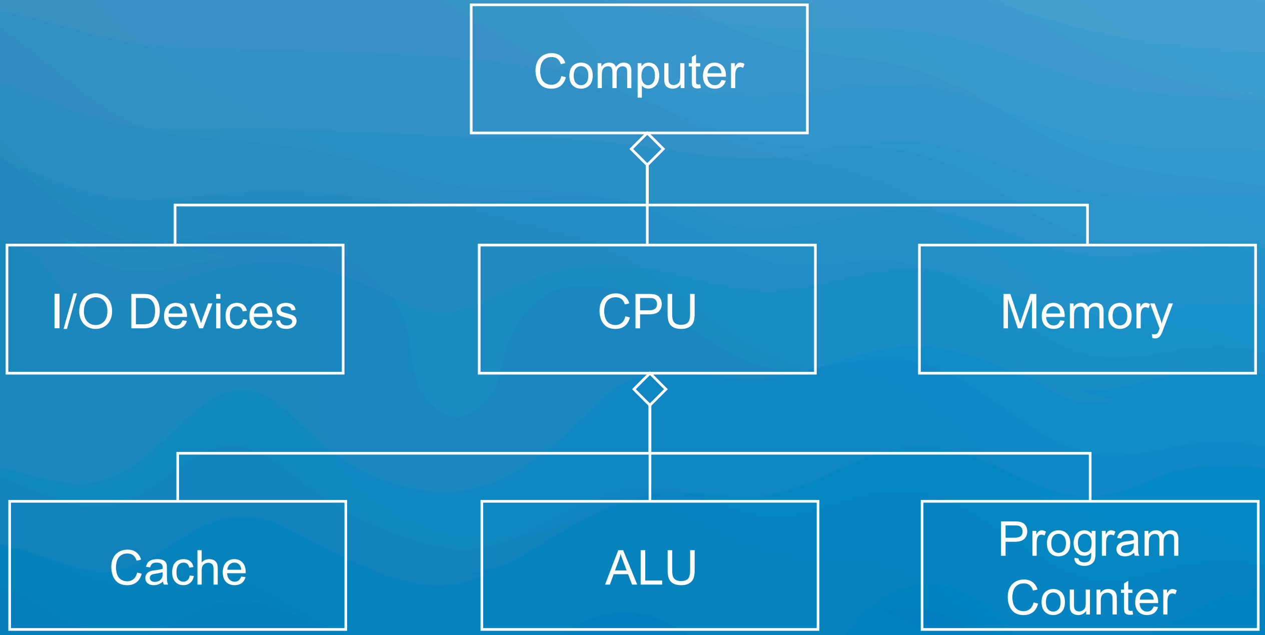
1-to-many association

Many-to-many Associations



- A stock exchange lists many companies.
- Each company is identified by a ticker symbol

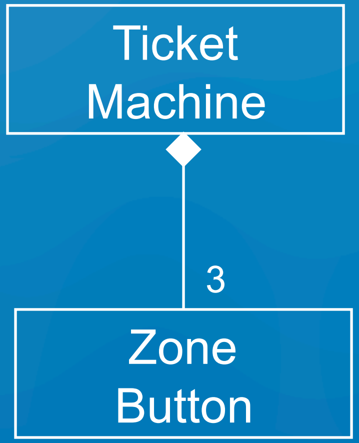
Part-of Hierarchy (Aggregation)



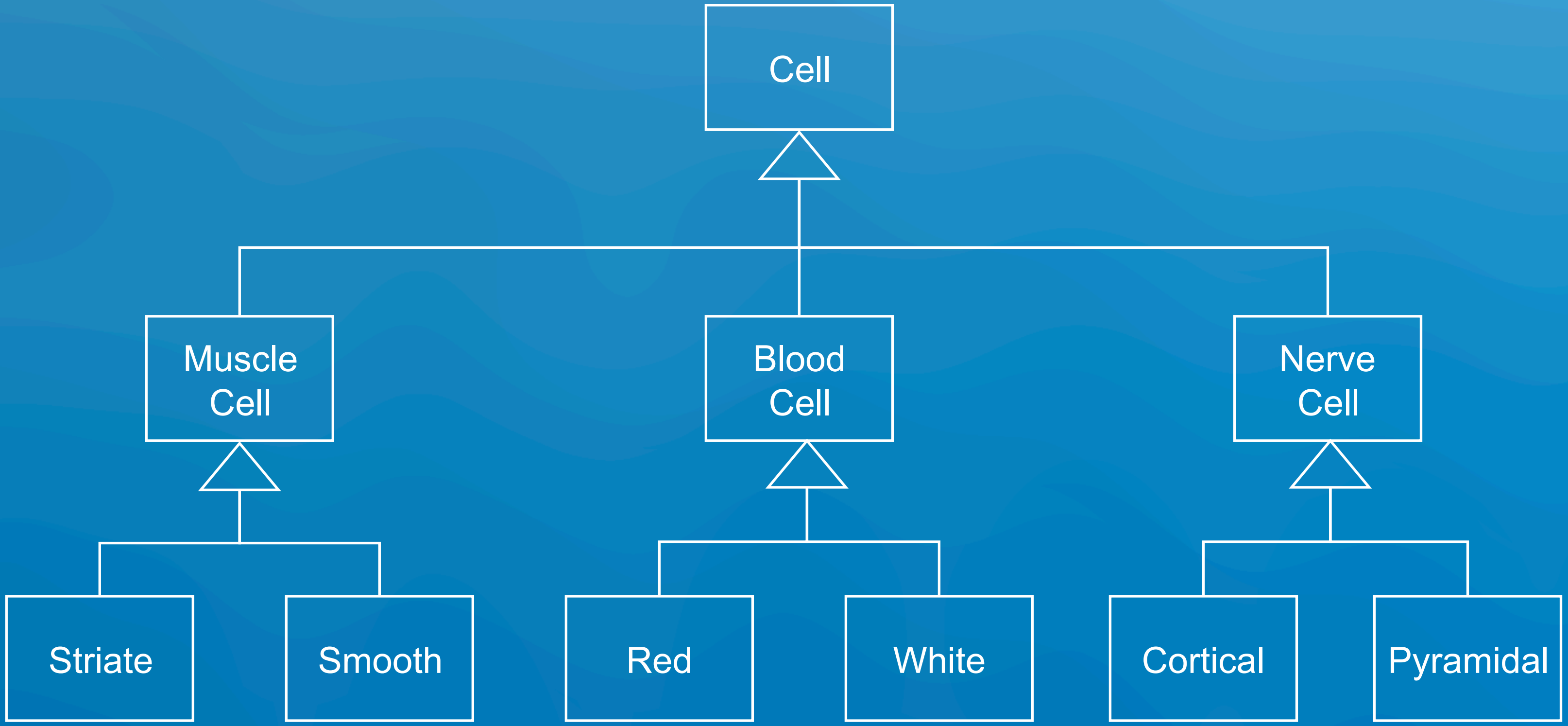
- An aggregation is a special case of association denoting a “consists-of” hierarchy
- The aggregate is the parent class, the components are the children classes

Composition

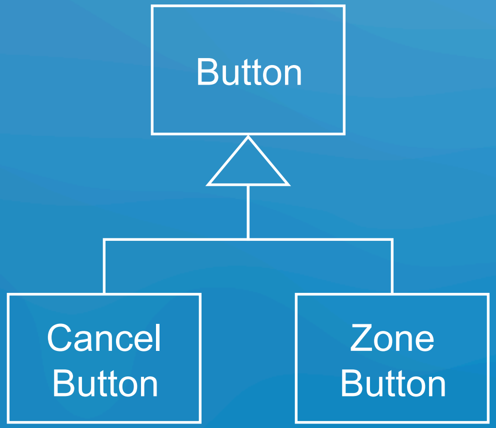
- A solid diamond denotes composition: A strong form of aggregation where the life time of the component instances is controlled by the aggregate (“the whole controls/destroys the parts”)



Is-Kind-of Hierarchy (Taxonomy)

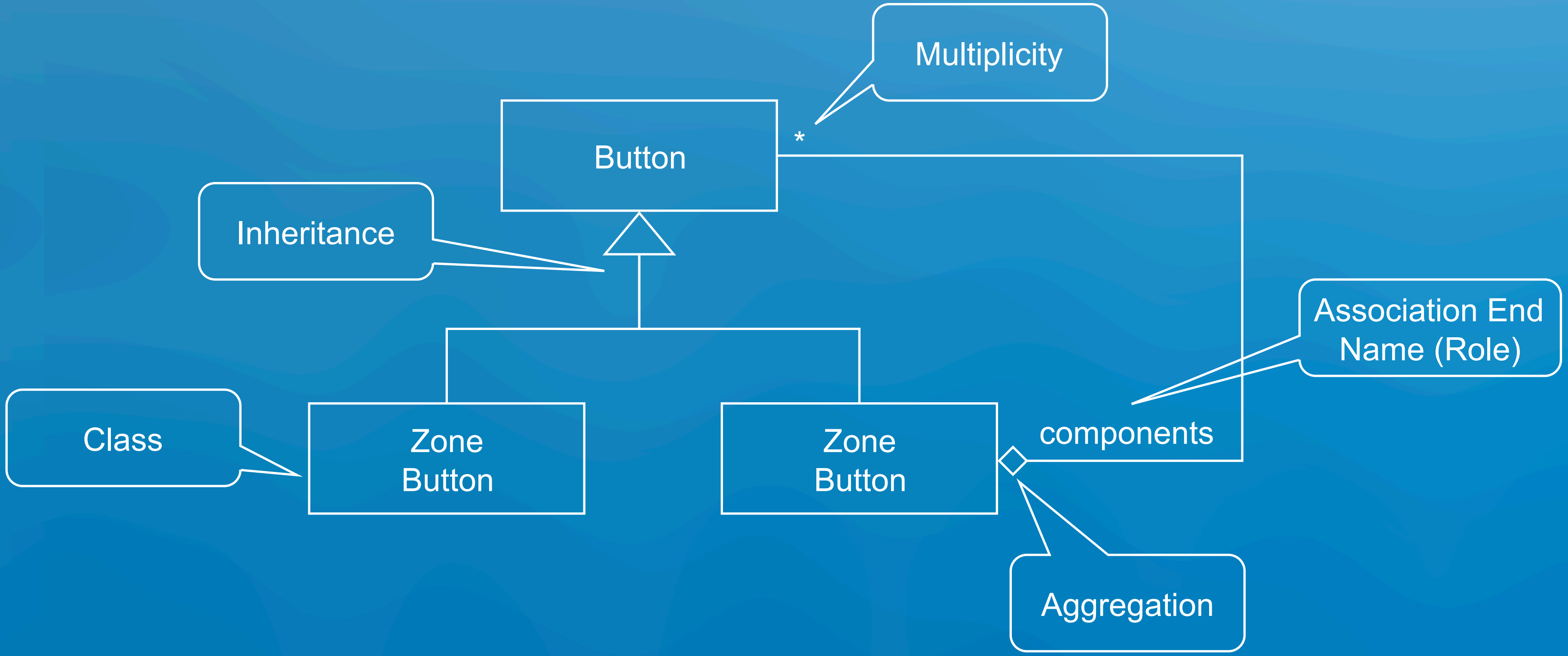


Inheritance



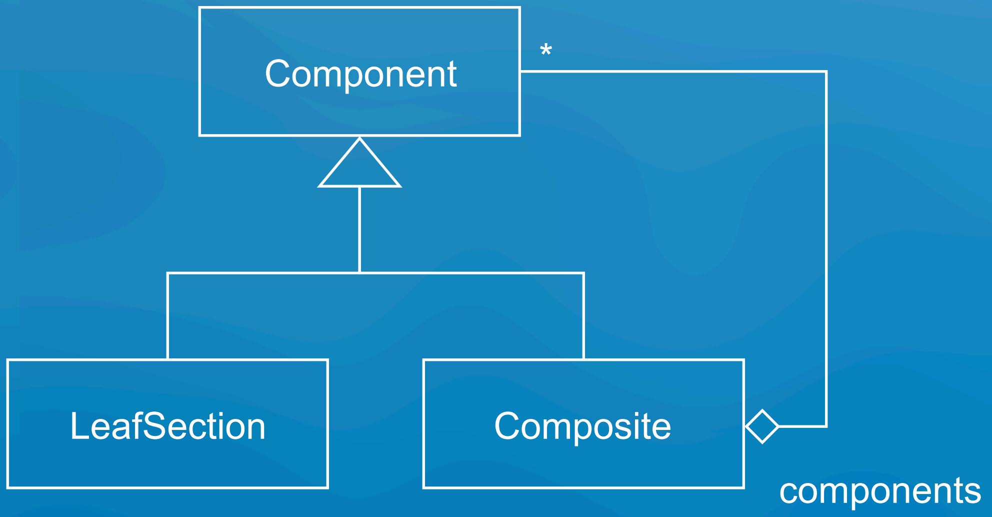
- ***Inheritance*** is another special case of an association denoting a “kind-of” hierarchy
- Inheritance simplifies the analysis model by introducing a taxonomy
- The **children classes** inherit the attributes and operations of the **parent class.**

Class diagram: Basic Notations



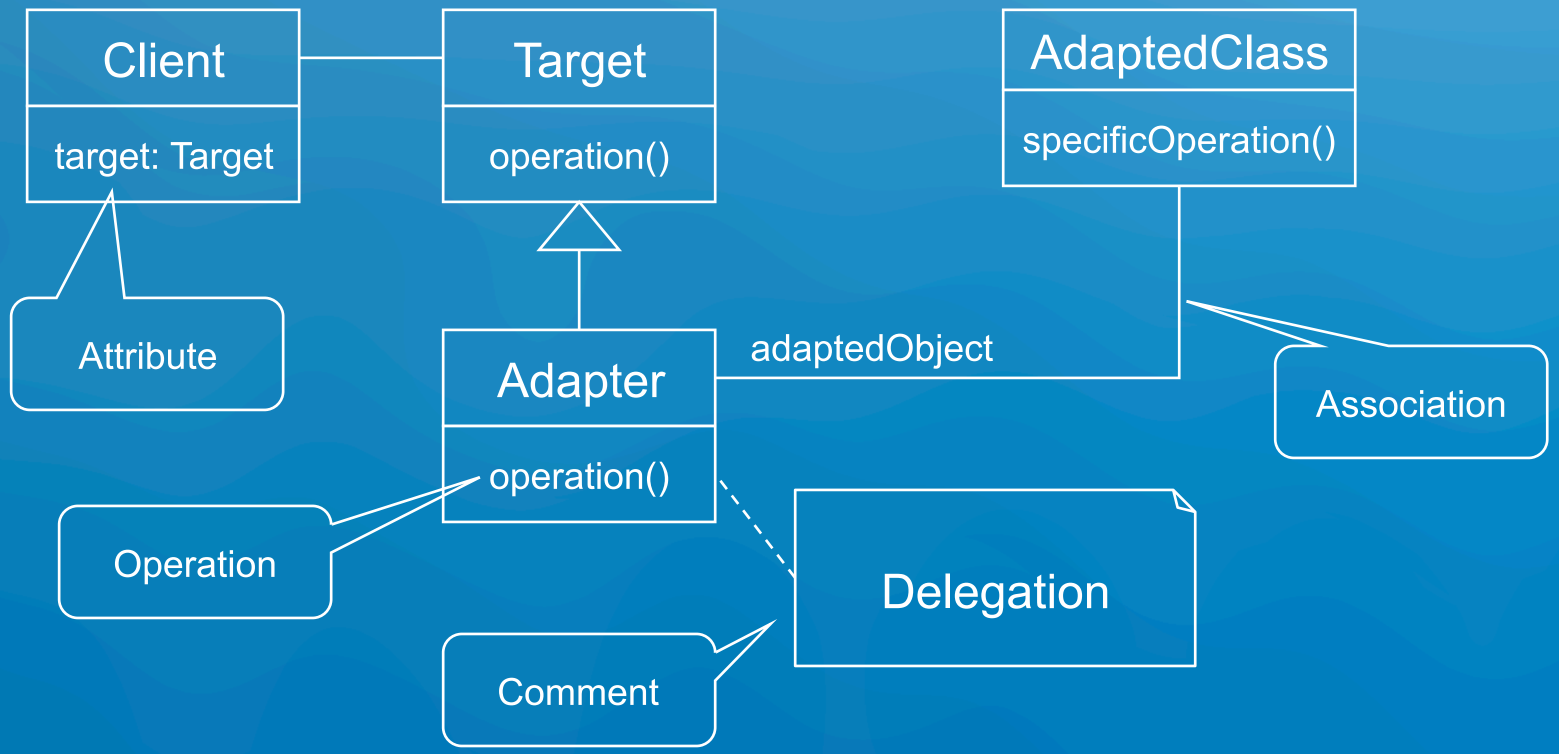
Class diagrams represent the structure of the system

Code Generation from UML to Java I

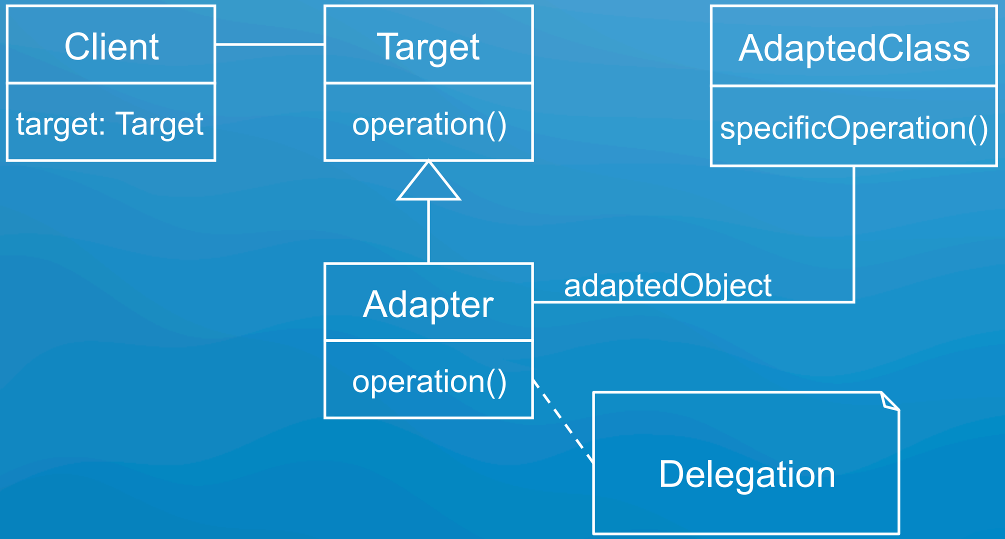


```
public class Component{  }

public class Leaf extends
   Component{  }

public class Composite extends
   Component{
   private Collection<Component>
   components;
   …
}
```

Class diagram: Basic Notations



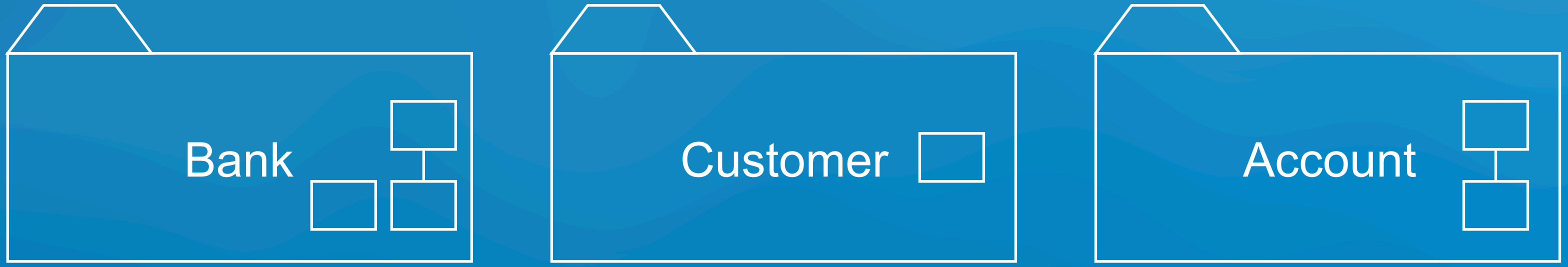
Code Generation from UML to Java II



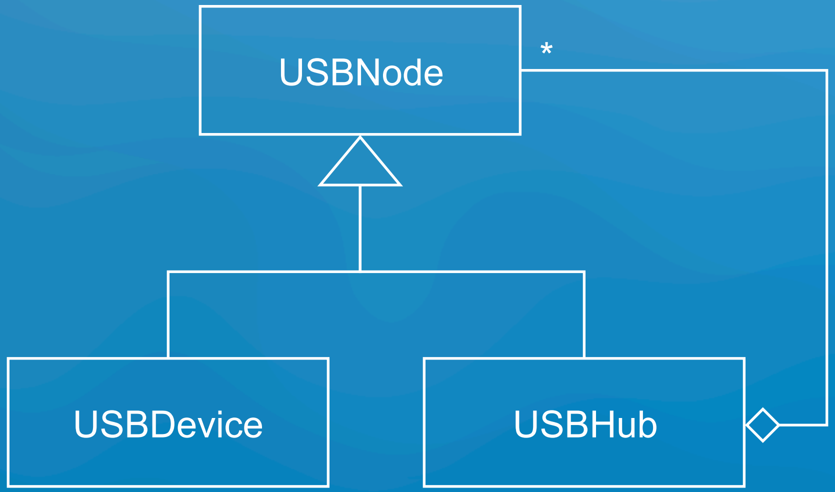
```
public abstract class Target{
   public … operation(); }

public class Adapter extends Target {
   private AdaptedClass adaptedObject;
   public … operation(){
        adaptedObject.specificOperation();
   }
```

Excursion: Packages

- Packages help you to organize UML models to increase their readability
- We can use the UML package mechanism to organize classes into subsystems



- Any complex system can be decomposed into subsystems, where each subsystem is modeled as a package.


USBNode
*
USBDevice
USBHub