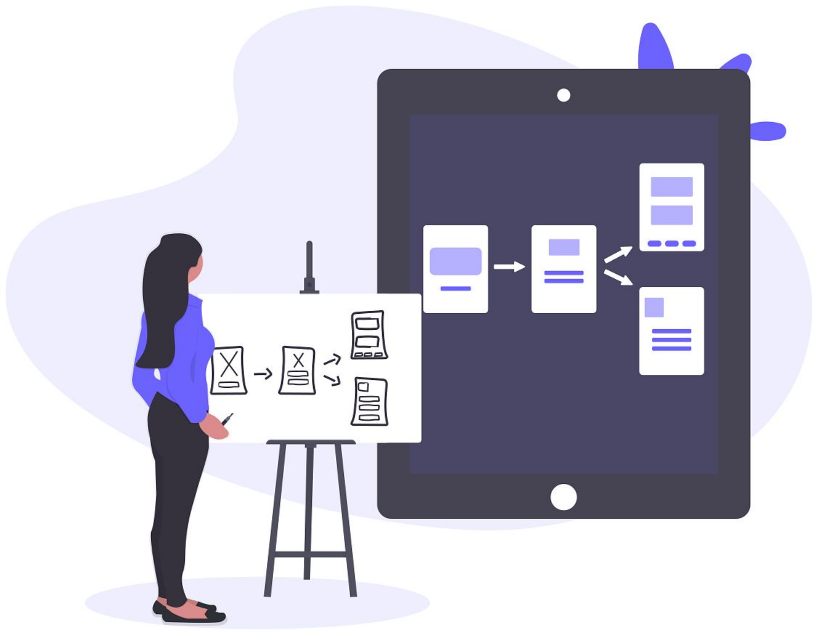
UML Basics

By Galia Georgieva



Agenda

- What is UML and why to use it
- Components of UML and diagram types
- Use Case Diagram + Exercise
- Class Diagram + Exercise
- Sequence Diagram + Exercise

What is UML and

Why to Use It?

What is UML?

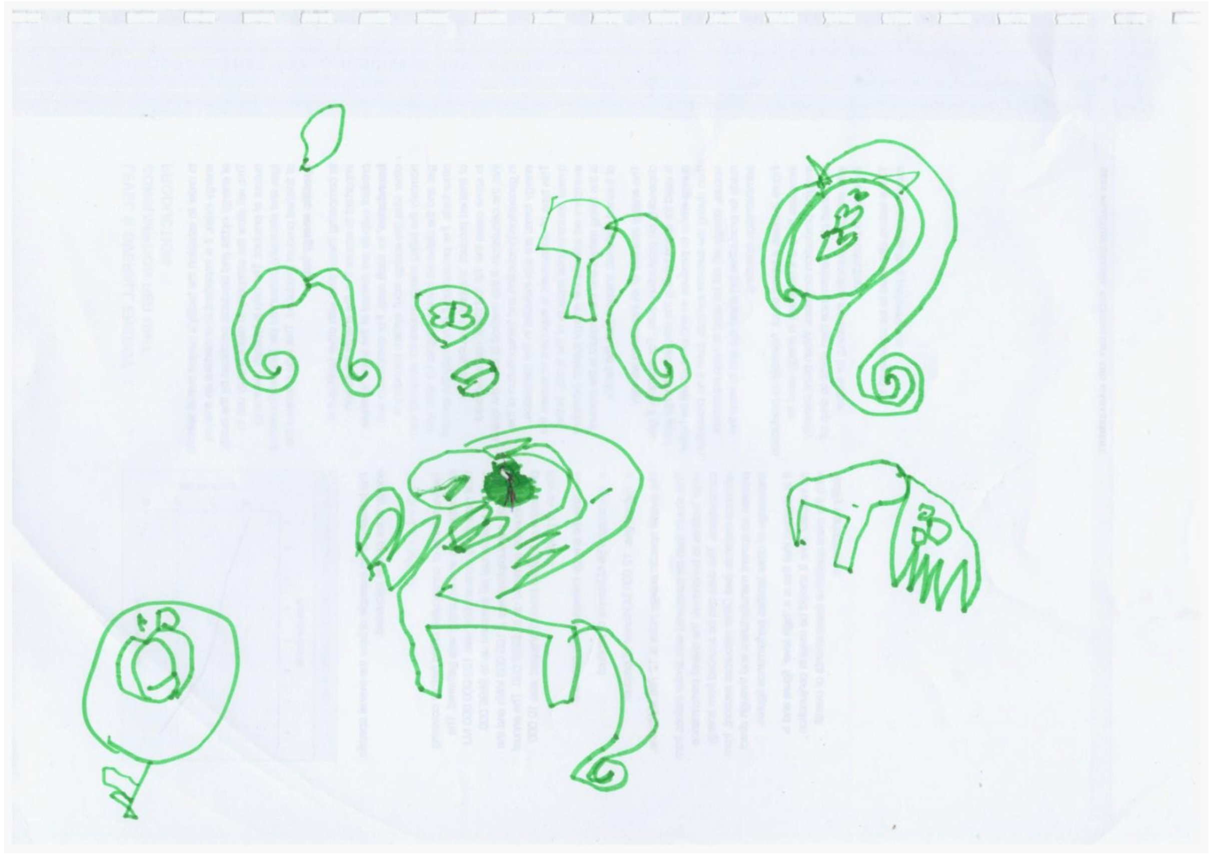
“A picture is worth a thousand words”

UNIFIED MODELING LANGUAGE™ UML®

UML = Unified Modeling Language

When you write a document, you want to convey information to someone with it. So it is important for you both to “talk” the same language.

Life without UNIFIED MODELING LANGUAGE



Model Driven Engineering Approach

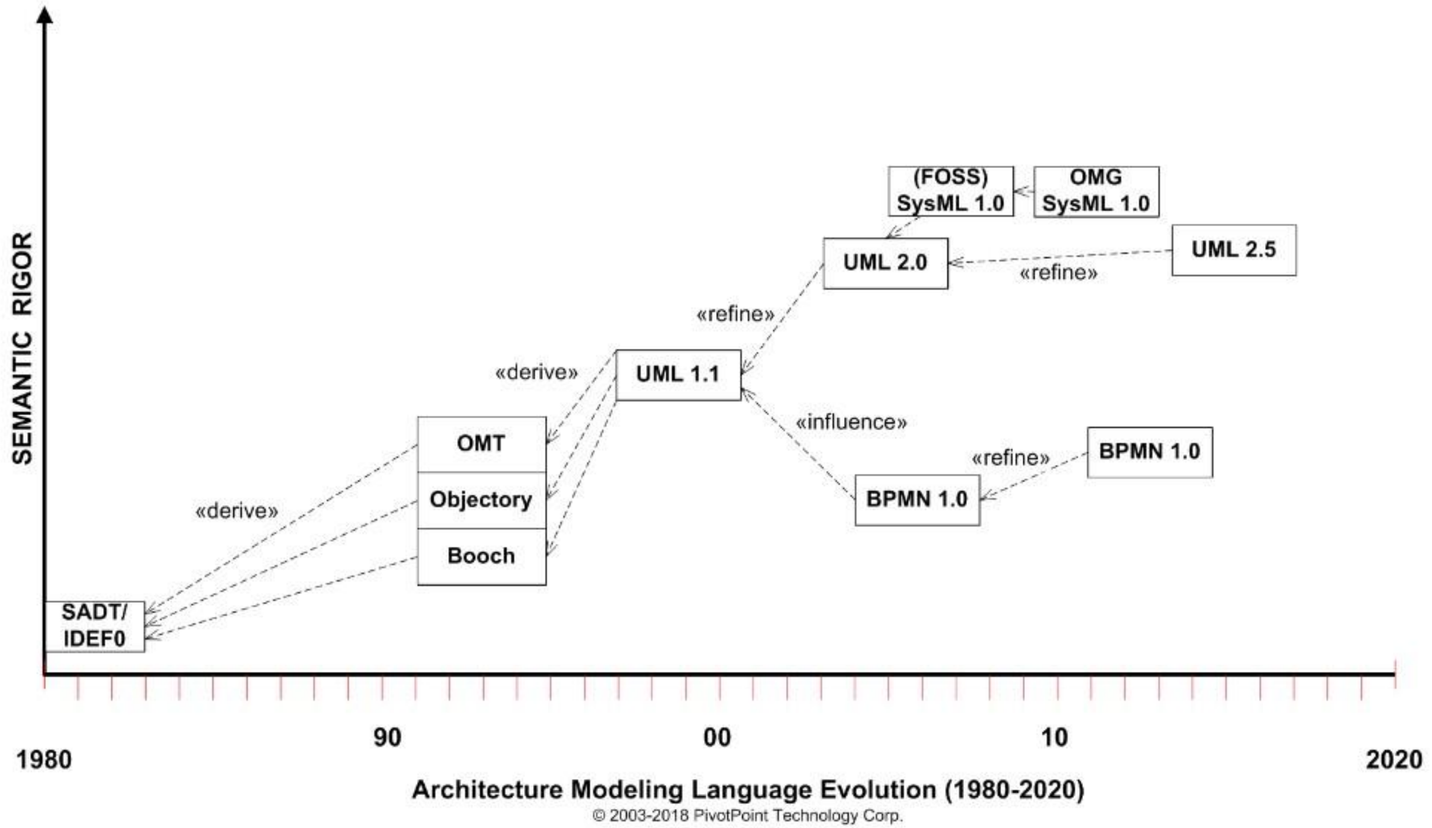
- **Reuse standard models:** Increase productivity and maximise compatibility between systems

- **Models of recurring design patterns in the application domain:** simplifying the process of design

- **Standardization of the terminology:** promoting communication between individuals and teams working on the system

Several variations of the modeling definitions were joined creating the Unified Modeling Language (UML).

History of UML

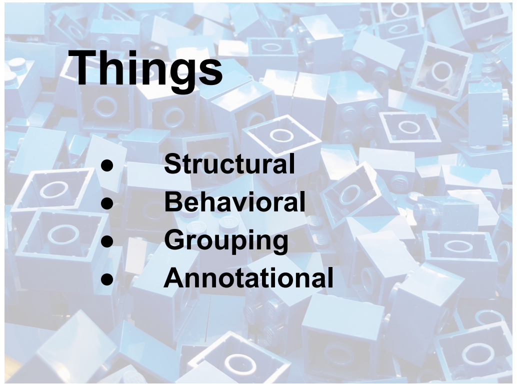


Why UML?

1. Provide users with a **ready-to-use**, expressive **visual modeling language** so they can develop and **exchange meaningful models**.
2. Provide **extensibility and specialization mechanisms** to extend the core concepts.
3. Be **independent** of particular programming languages and development processes.
4. Provide **a formal basis for understanding** the modeling language.
5. **Support higher-level development concepts** such as collaborations, frameworks, patterns and components.
6. Integrate **best practices.**

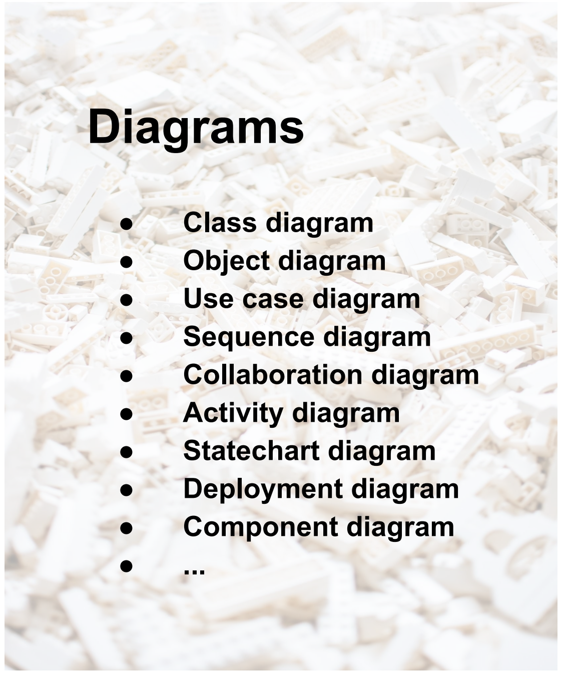
Components of UML and Diagram Types

UML building blocks



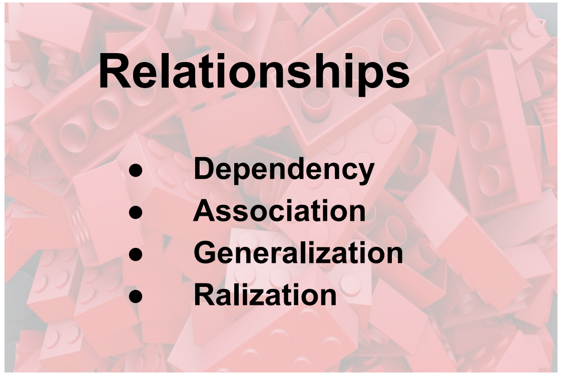
Things

- Structural
- Behavioral
- Grouping
- Annotational



Diagrams

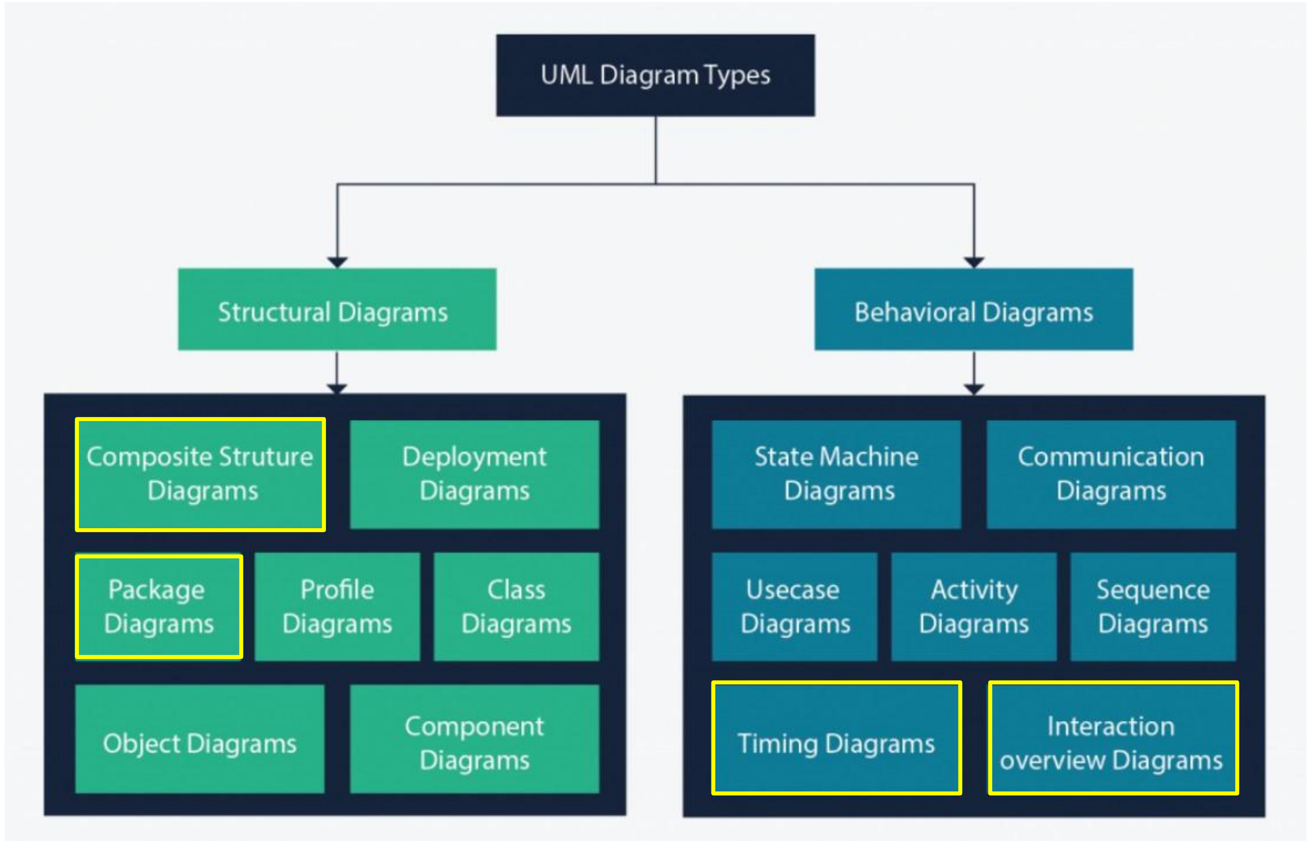
- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Collaboration diagram
- Activity diagram
- Statechart diagram
- Deployment diagram
- Component diagram
- ...



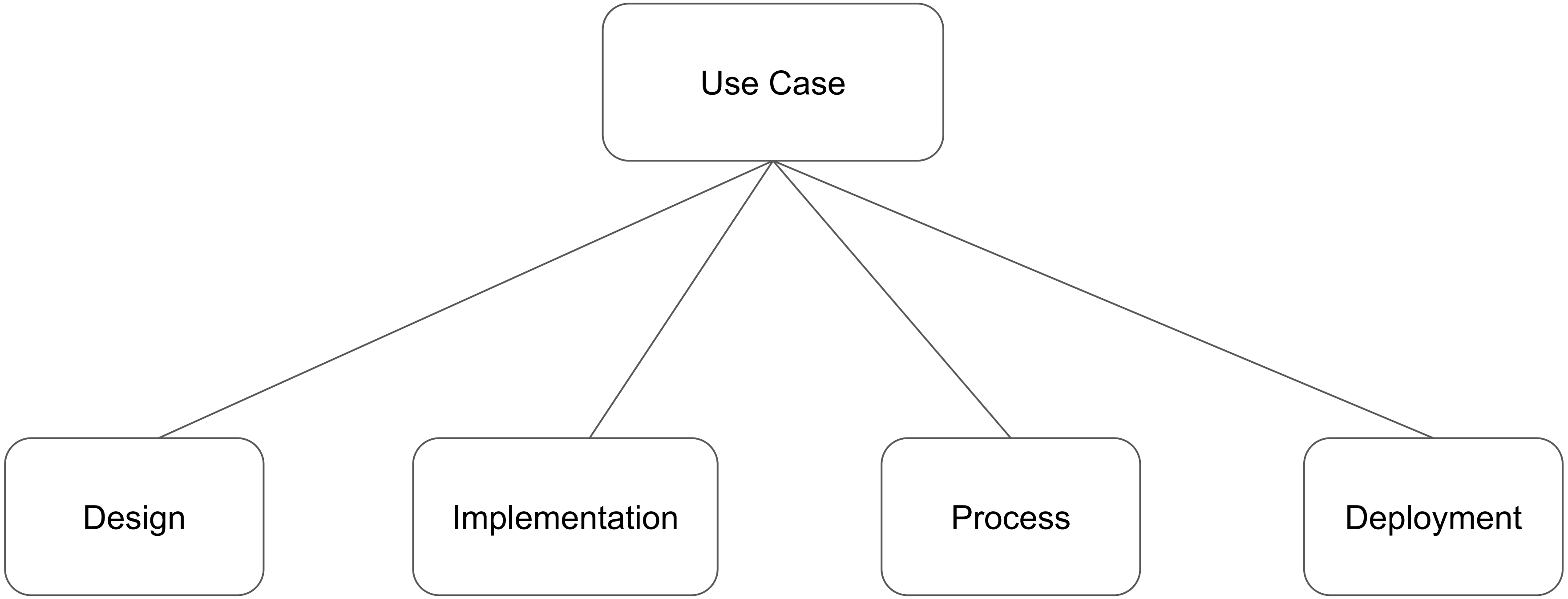
Relationships

- Dependency
- Association
- Generalization
- Ralization

Diagrams and modeling types



Using UML to define different perspectives of a system



Use Case Diagram

Use Case Diagram

Purpose of Use Case Diagrams

1. Specify the context of a system
2. Capture the requirements of a system
3. Validate a systems architecture
4. Drive implementation and generate test cases
5. Developed by analysts together with domain experts

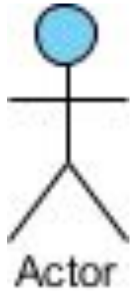
Components of an UML Use Case Diagram

- Actors
- Use Cases
- Communication Links
- System Boundaries

Notation Description

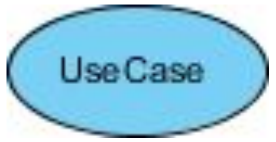
Actor

Someone interacts with use case (system function).
Named by noun.
Actor plays a role in the business
Similar to the concept of user, but a user can play different roles



Use Case

System function (process - automated or manual)
Named by verb + Noun (or Noun Phrase).
i.e. Do something
Each Actor must be linked to a use case, while some use cases may not be linked to actors.



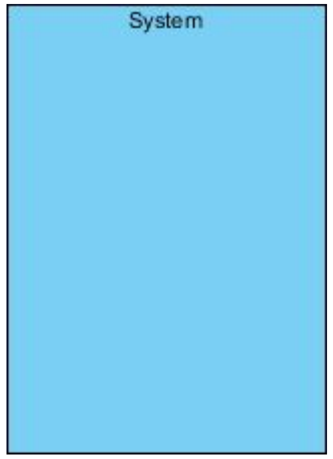
Notation Description

Communication Link

The participation of an actor in a use case is shown by connecting an actor to a use case by a solid link.
Actors may be connected to use cases by associations, indicating that the actor and the use case communicate with one another using messages.

Boundary of system

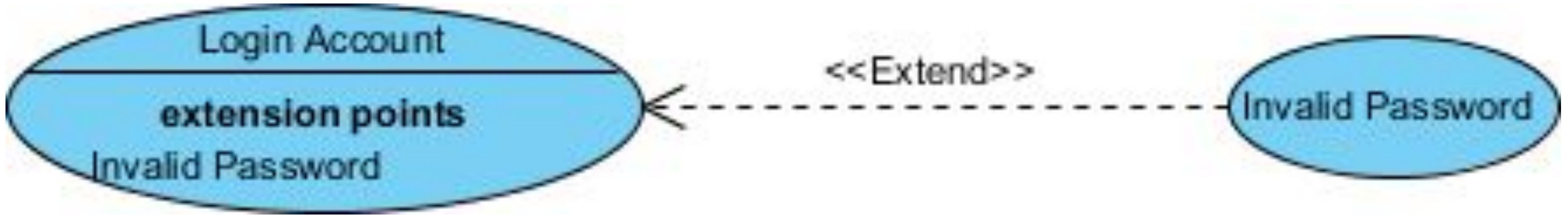
The system boundary is potentially the entire system as defined in the requirements document.
For large and complex systems, each module may be the system boundary.



Use Case Relationships

Extends

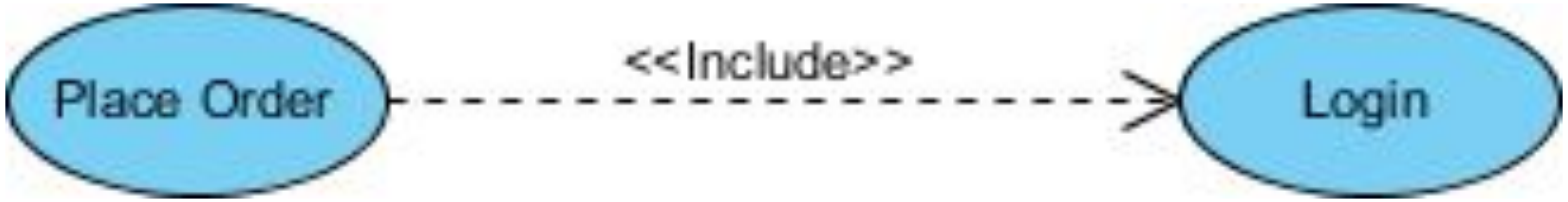
- Indicates that an use case **may include** the behavior specified by base use case.
- The tip of arrowhead points to the base use case and the child use case is connected at the base of the arrow.
- The stereotype "<<extends>>" identifies as an extend relationship



Use Case Relationships

Include

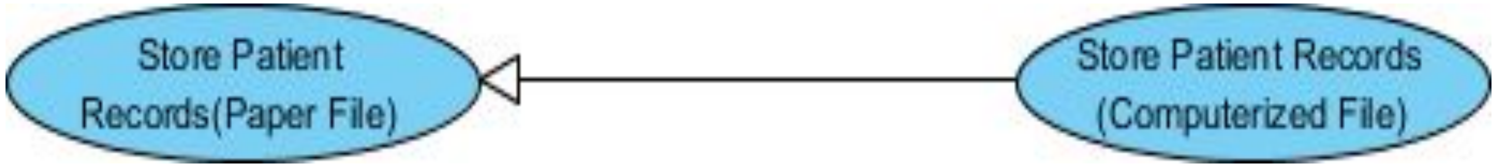
- A use case includes the functionality described in another use case as a part of its business process flow.
- A uses relationship from base use case to child use case indicates that an instance of the base use case **will include** the behavior as specified in the child use case.
- The tip of arrowhead points to the child use case and the parent use case connected at the base of the arrow.
- The stereotype "<<include>>" identifies the relationship as an include relationship.



Use Case Relationships

Generalization

- A generalization relationship is a parent-child relationship between use cases.
- The child use case is an enhancement of the parent use case.
- Generalization is shown as a directed arrow with a triangle arrowhead.
- The child use case is connected at the base of the arrow. The tip of the arrow is connected to the parent use case.


Store Patient Records(Paper File)
Store Patient Records (Computerized File)

Example - Create Use Case Diagram

Requirements:

As a customer, I want to be able to place an order. The order should hold information about the date received, whether it's currently prepared, and a price.

As a shopkeeper I want to be able to receive payments from corporate and individual customers. I want to be able to accept payments from individual customers credit cards, and for the corporate customers, I want to be able to check the credit rating and the credit limit, for the corporate contact.

Example - Create Use Case Diagram

Class Diagram

Class Diagram

Purpose of Class Diagrams

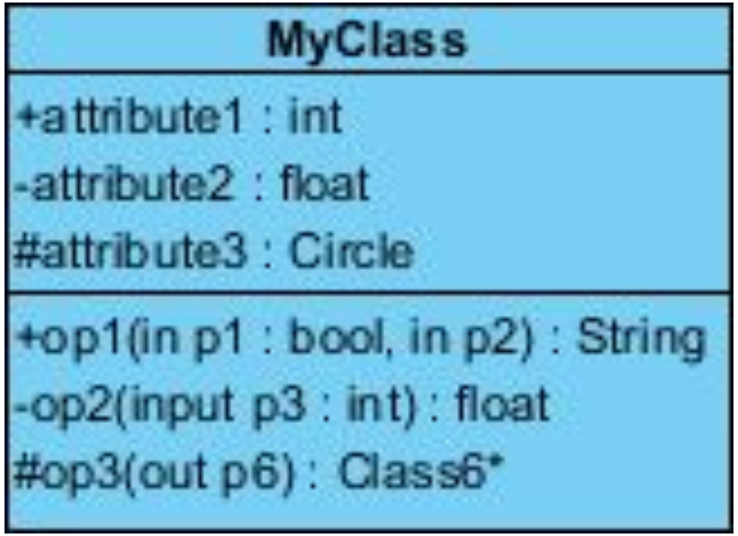
1. Shows static structure of classifiers in a system
2. Diagram provides a basic notation for other structure diagrams prescribed by UML
3. Helpful for developers and other team members too
4. Business Analysts can use class diagrams to model systems from a business perspective

Components of an UML Class Diagram

- A set of classes and
- A set of relationships between classes

Class Notation

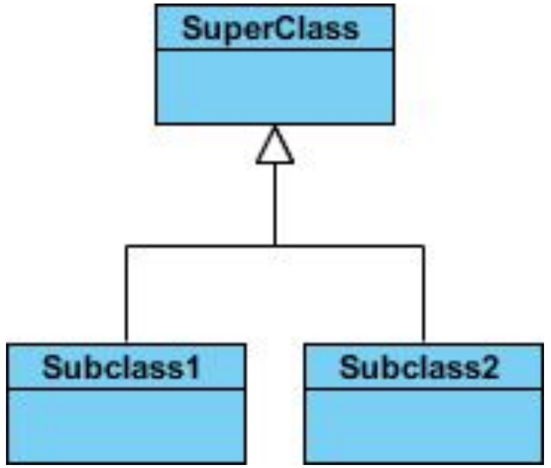
1. **Class Name**
   - The name of the class appears in the first partition.
2. **Class Attributes**
   - Attributes are shown in the second partition.
   - The attribute type is shown after the colon.
   - Attributes map onto member variables (data members) in code.
3. **Class Operations** (Methods)
   - Operations are shown in the third partition. They are services the class provides.
   - The return type of a method is shown after the colon at the end of the method signature.
   - The return type of method parameters is shown after the colon following the parameter name.
   - Operations map onto class methods in code



Class Relationships

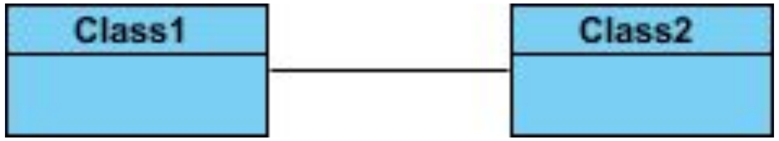
Inheritance (or Generalization):

- Represents an "is-a" relationship.
- An abstract class name is shown in italics.
- SubClass1 and SubClass2 are specializations of Super Class.
- A solid line with a hollow arrowhead that point from the child to the parent class



Simple Association:

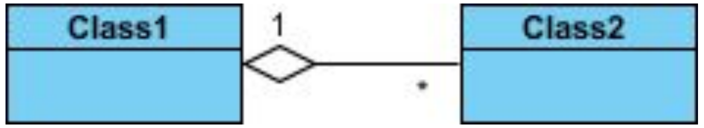
- A structural link between two peer classes.
- There is an association between Class1 and Class2
- A solid line connecting two classes



Class Relationships

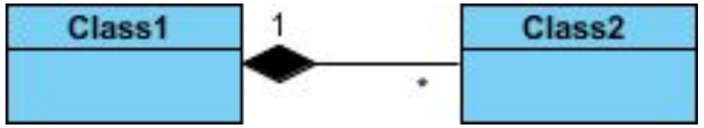
Aggregation:

A special type of association. It represents a "part of" relationship.



- Class2 is part of Class1.
- Many instances (denoted by the *) of Class2 can be associated with Class1.
- Objects of Class1 and Class2 have separate lifetimes.
- A solid line with an unfilled diamond at the association end connected to the class of composite

Composition:
A special type of aggregation where parts are destroyed when the whole is destroyed.

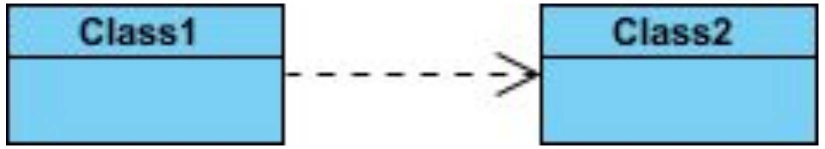


- Objects of Class2 live and die with Class1.
- Class2 cannot stand by itself.
- A solid line with a filled diamond at the association connected to the class of composite

Class Relationships

Dependency:

- Exists between two classes if the changes to the definition of one may cause changes to the other (but not the other way around).
- Class1 depends on Class2
- A dashed line with an open arrow



Exercise - Create a class diagram

Requirements:

As a customer, I want to be able to place an order. The order should hold information about the date received, whether it's currently prepared, and a price.

As a shopkeeper I want to be able to receive payments from corporate and individual customers. I want to be able to accept payments from individual customers credit cards, and for the corporate customers, I want to be able to check the credit rating and the credit limit, for the corporate contact.

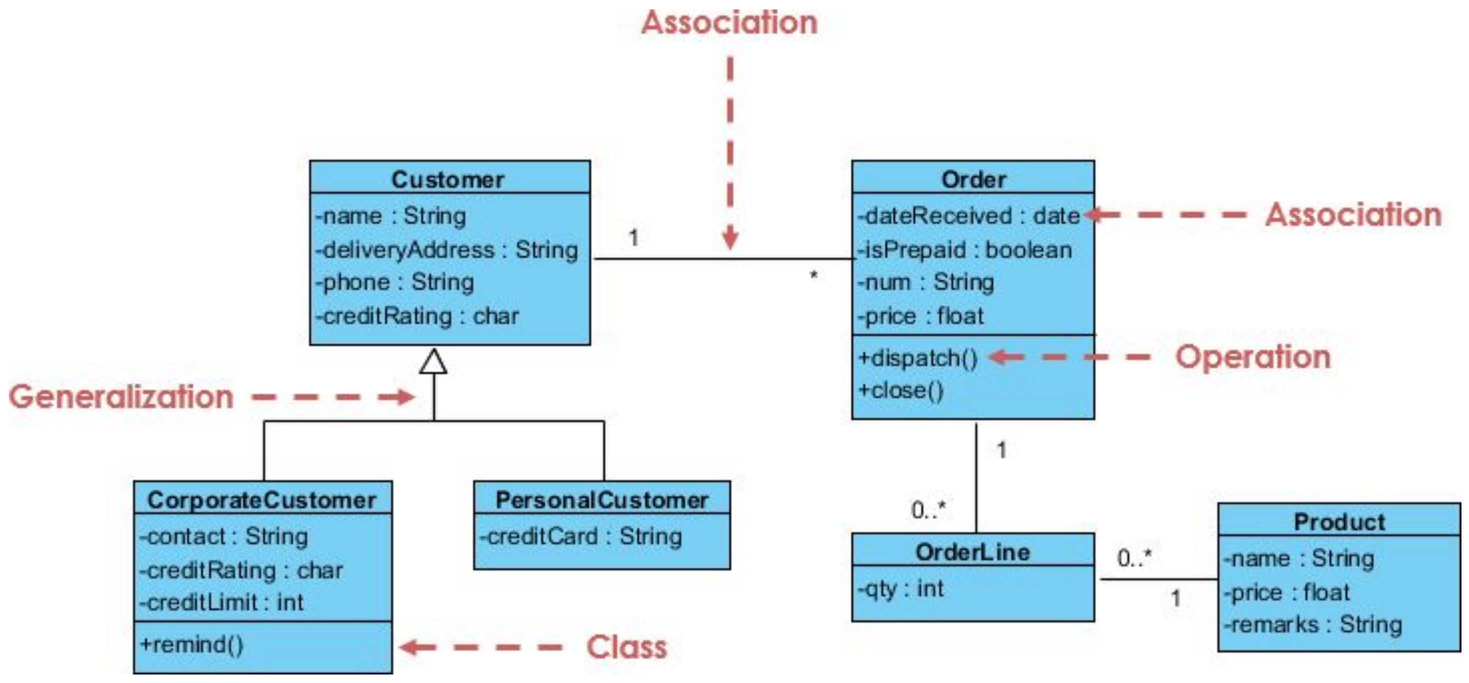
I want to be able to remind a payment is pending to the corporate customer.

As a shopkeeper, I want to be able to dispatch and close an order.

For each order, I need to have multiple order lines with quantity, for the different products.

A product should hold information about name, price and remarks.

Class Diagram Example



Sequence Diagram

Sequence Diagram

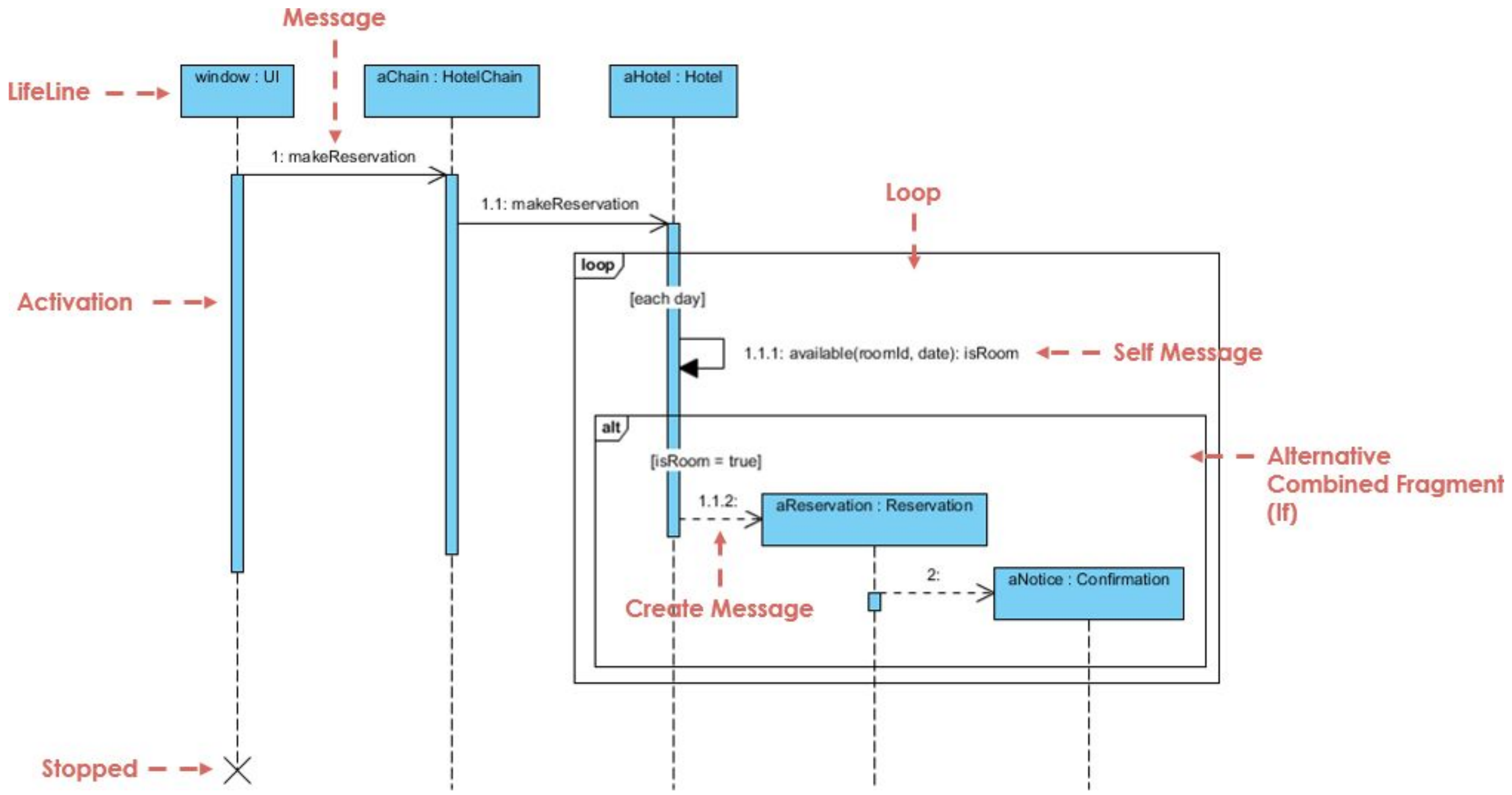
Purpose of Class Diagrams

1. Model high-level interaction between active objects in a system
2. Model the interaction between object instances within a collaboration that realizes a use case
3. Model the interaction between objects within a collaboration that realizes an operation
4. Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction)

Components of an UML Class Diagram

- Object Dimension
- Time Dimension

Sequence Diagram Example



Questions?