UML REVIEW

We will use UML this term

Modelling language consisting of set of diagrams

Used to help developers to specify, visualize, construct, and document software system

1. Class diagrams
2. Collaboration diagrams
3. Sequence diagrams

Let’s review them

Class diagram

Purpose is to describe the static structure of the system

Shows the classes

- Attributes
- Operations

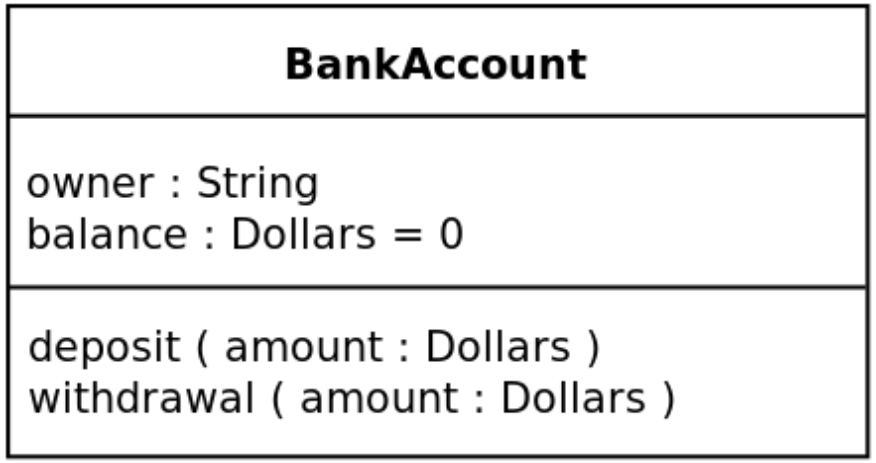
Relationships among objects

Language agnostic

Class diagram

Rectangle indicate a class

- Top section is the name of the class
- Middle section are the attributes and types
- Bottom section are the methods



Class diagram - Relationships

Arrows indicate how two classes are related

Association – Reference based relationship between two classes. A contains member variables B

Generalization/Inheritance – Indicates that one of the two related classes is a specialized form of the other. A is a subclass of B

Class diagram - Relationships

Arrows indicate how two classes are related

Realization/Implementation – A relationship between two classes where one class implements the behaviors that the other class specifies. A is implementing interface B

A – – –▷ B

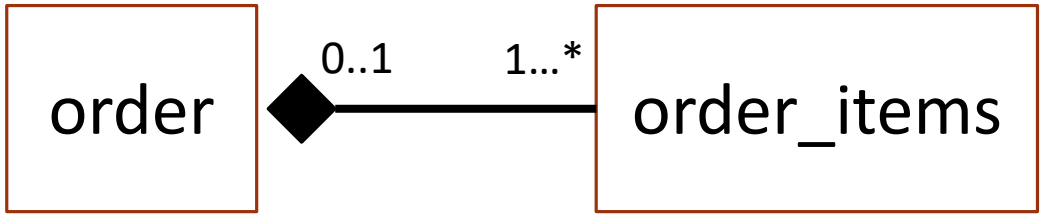
Dependency – Communication between two classes, where one class does not necessarily contain an instance of another class. A contains methods that use B

A – – – → B

Class diagram

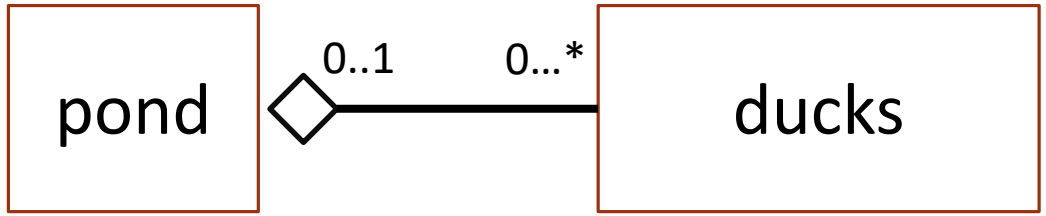
Composition: When a class is made up of and has references to objects of another class. In a composition relationship, the parts cannot exist without the whole.

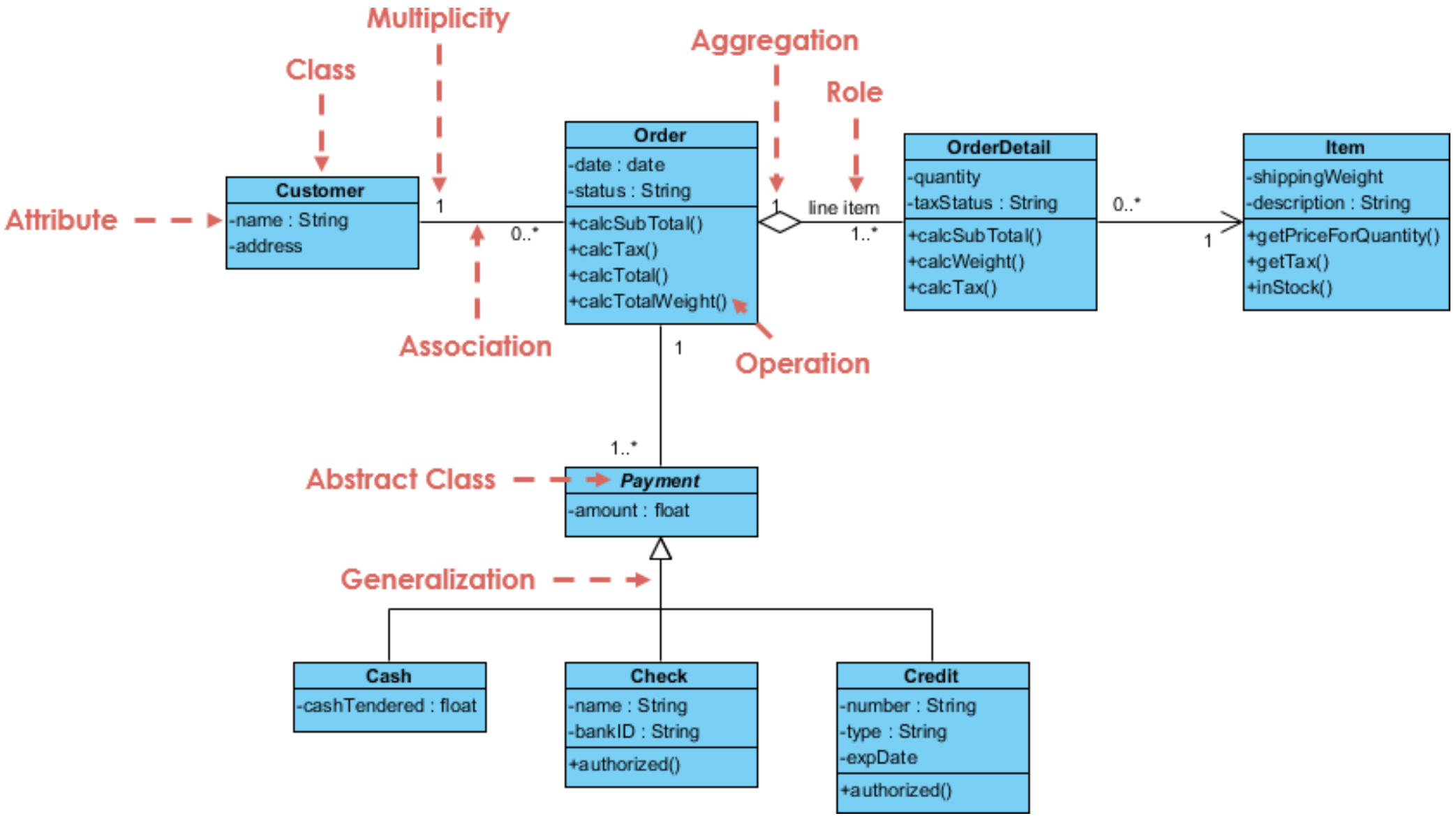
Eg: If an *order* is destroyed, then so are the *order_items* that are composed by *order*



Aggregation: When a class references an object of another class. In an aggregation relationship, the referenced class can live on even if the class referencing it is destroyed.

Eg: If a *Pond* is destroyed, the ducks can live on.

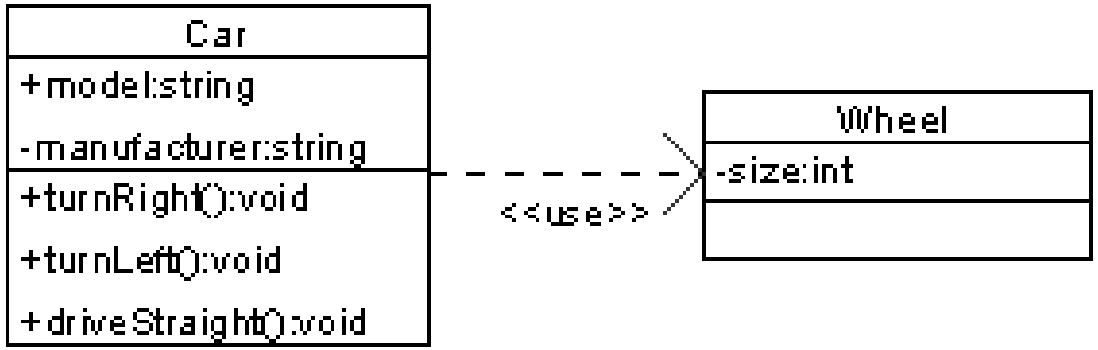



Multiplicity
Class
Aggregation
Role
Attribute
Customer
-name : String
-address
1
0..*
Order
-date : date
-status : String
+calcSubTotal()
+calcTax()
+calcTotal()
+calcTotalWeight()
1
line item
1..*
OrderDetail
-quantity
-taxStatus : String
+calcSubTotal()
+calcWeight()
+calcTax()
0..*
1
Item
-shippingWeight
-description : String
+getPriceForQuantity()
+getTax()
+inStock()
Association
Operation
1
1..*
Abstract Class
Payment
-amount : float
Generalization
Cash
-cashTendered : float
Check
-name : String
-bankID : String
+authorized()
Credit
-number : String
-type : String
-expDate
+authorized()

Class diagram

Car is composed of carburetors. Car has exactly 1 carburetor. Carburetor can belong to 0 to 1 cars

Class diagram

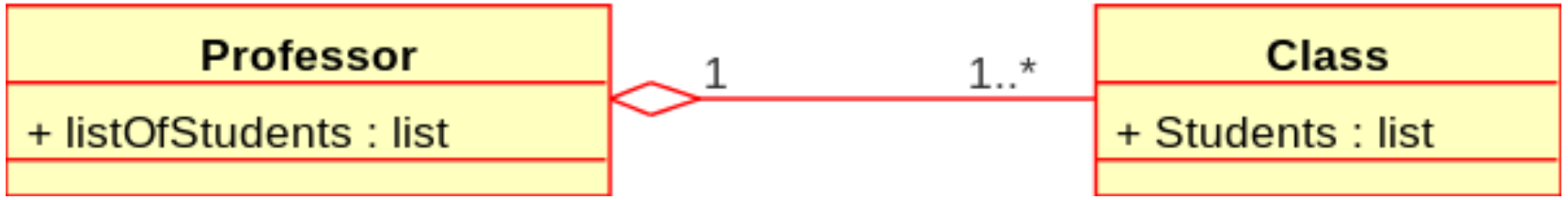


Car

- 2 strings named model, manufacturer
- Methods to turnRight, turnLeft, driveStraight

Car has a relationship with wheel. Some method uses wheel as a local variable to parameter

Class diagram



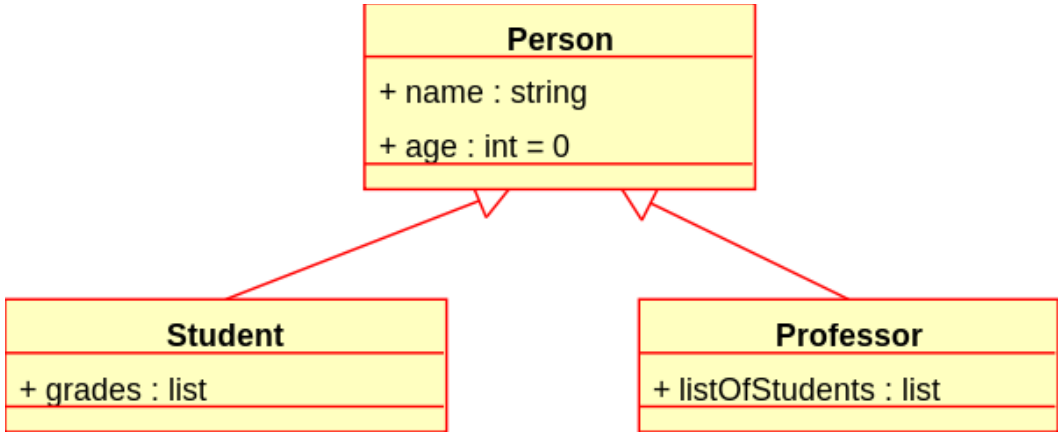
Professor is an aggregate of classes. Professor has 1 to many classes

- Has a list named listOfStudents

Each class has exactly 1 Professor

- Has a list named Students

Class diagram



Person class has a string name and int age as attributes

Student is a subclass of Person

- Has a list named grades

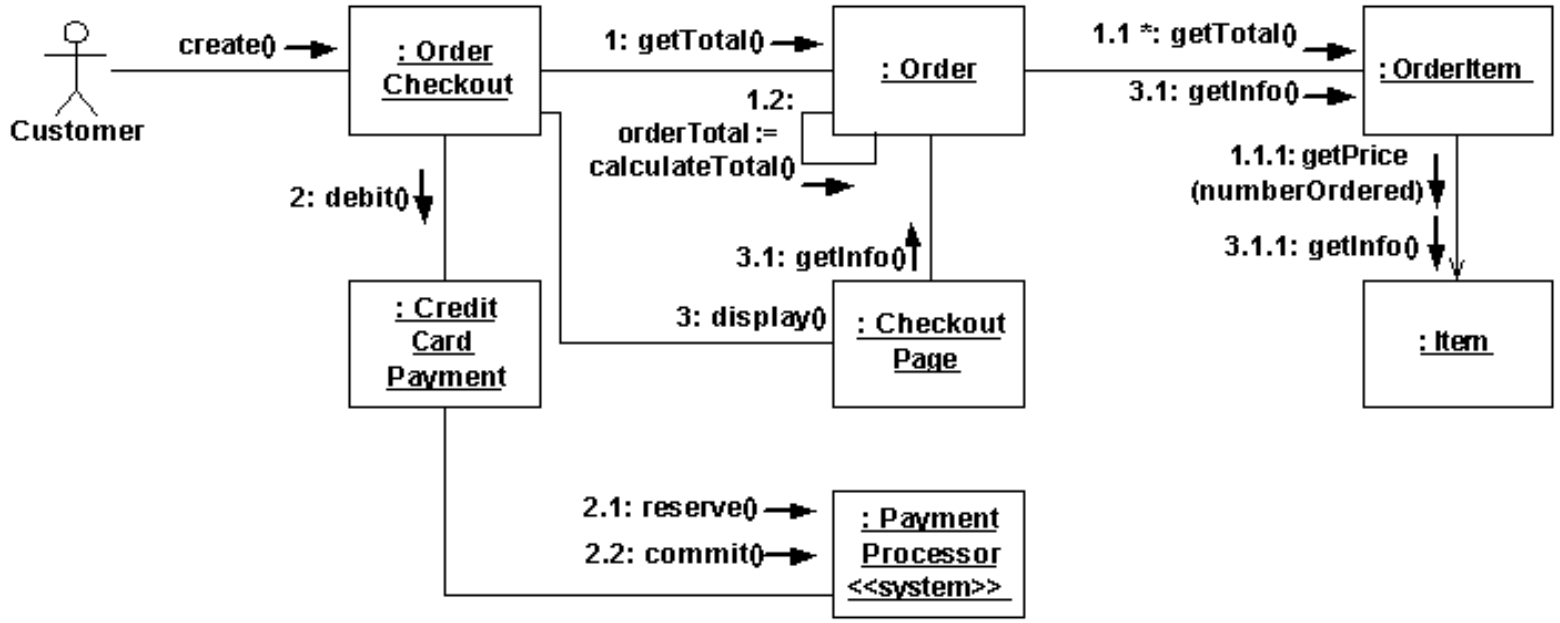
Professor is a subclass of Person

- Has a list named listOfStudents

Collaboration (communication) diagram

Presents interactions between objects as a sequence of messages between objects from a **structural perspective**

Shows how objects interact in a particular use case or subset of a use case



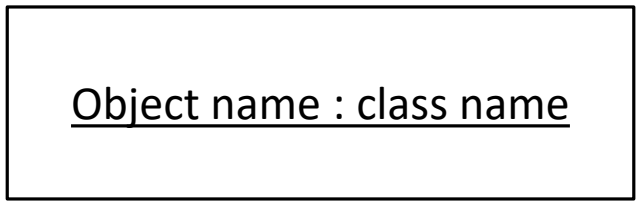
https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml-collaboration-diagram/

Collaboration (communication) diagram

Notations (Object, Actor, Links, Messages)

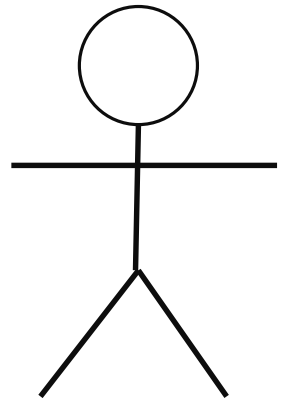
Object

- Object represented by a square containing the name of the object and its class underlined, separated by a colon



Actor

- Actor is the invoker of the interaction
- Actor is named and has a role



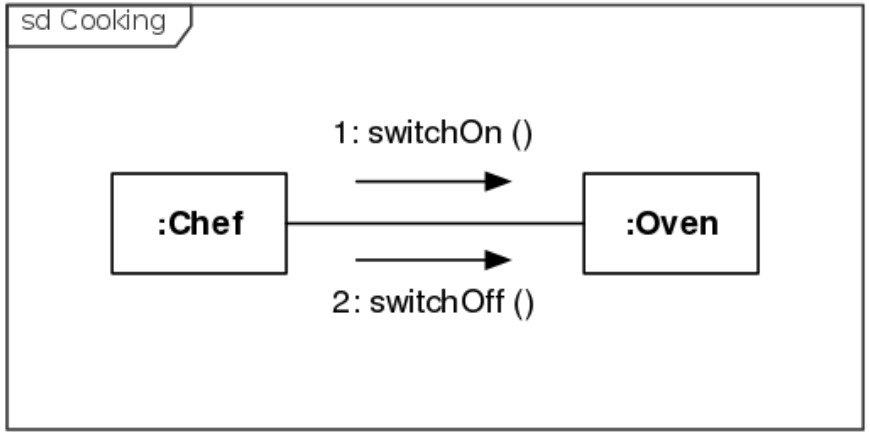
Customer

Collaboration (communication) diagram

Notations (Object, Actor, Links, Messages)

Links

- Connects objects and actors. They are instances of associations in the class diagram
- Messages are sent across these links
- Links are represented as a **solid line** between two objects

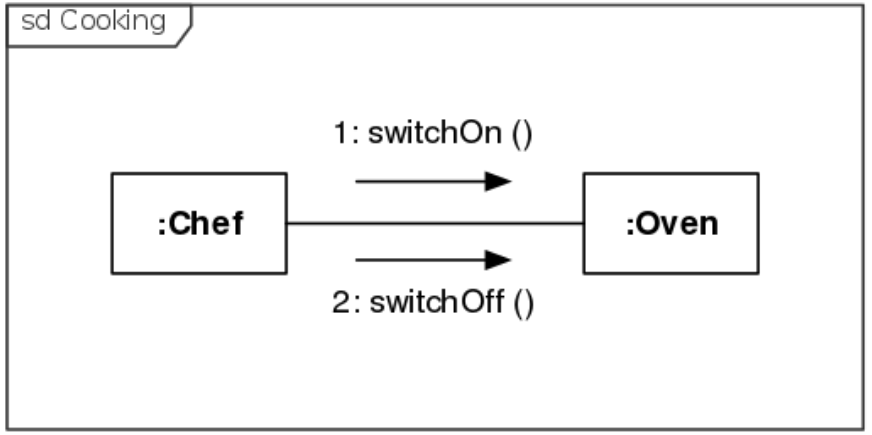


Collaboration (communication) diagram

Notations (Object, Actor, Links, Messages)

Messages

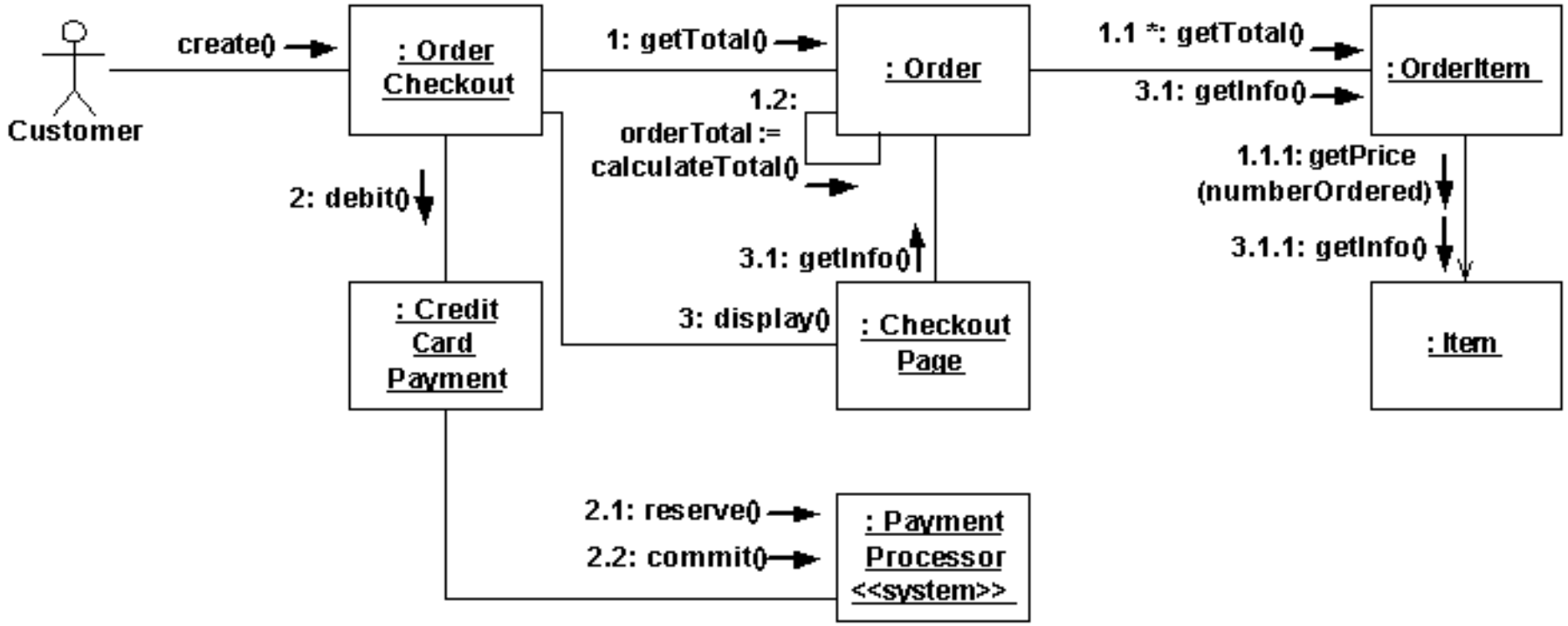
- Communication between objects that convey information with the expectation that activity occurs
- It is shown as a **labeled arrow** near a link
- Directed from sender to receiver
- Messages are numbered to indicate a sequence. Messages in the same call have additional decimals: 1.1, 1.2 etc



Collaboration (communication) diagram

Let's see this diagram again

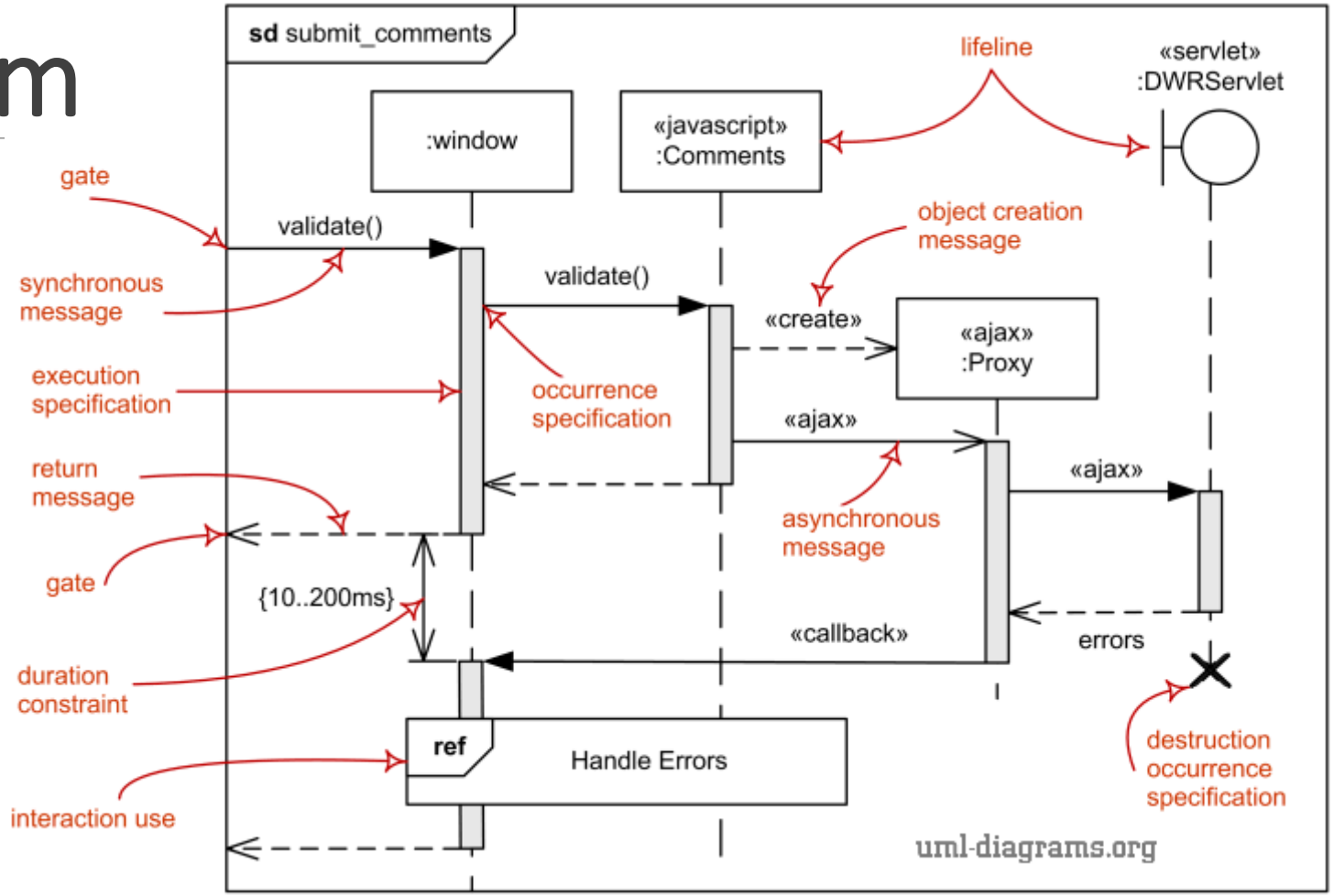
What use case is this collaboration diagram illustrating?



Sequence diagram

Presents **interactions between objects** as a sequence of messages between objects

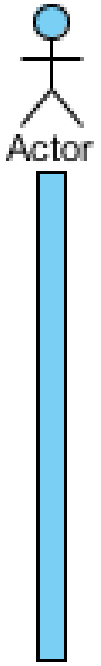
While a collaboration diagram is a structural approach, A sequence diagram is a **time-based perspective** includes the **lifetime** of the interactions between communicating objects



Sequence diagram

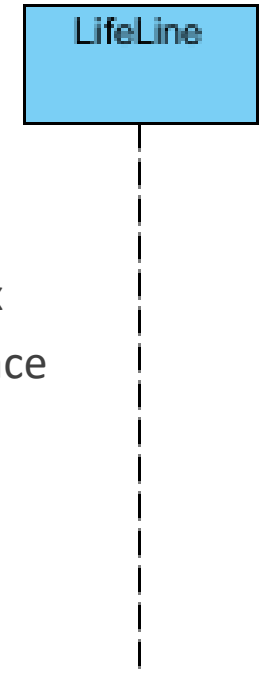
Actor

- Interacts with the subject
- External to the subject
- Played by human users, external hardware, other subjects



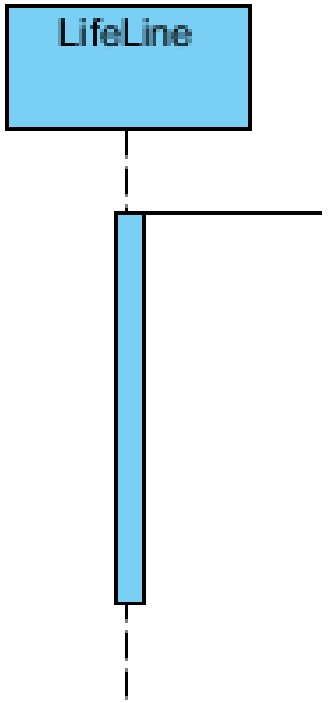
Lifeline

- Represents a single individual participant in the interaction
- Denoted by name and class type in the top box
- Dotted line indicates place in sequence



Activations

- Thin rectangle on the lifeline
- Represents the duration an element is performing an operation



https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/

Sequence diagram

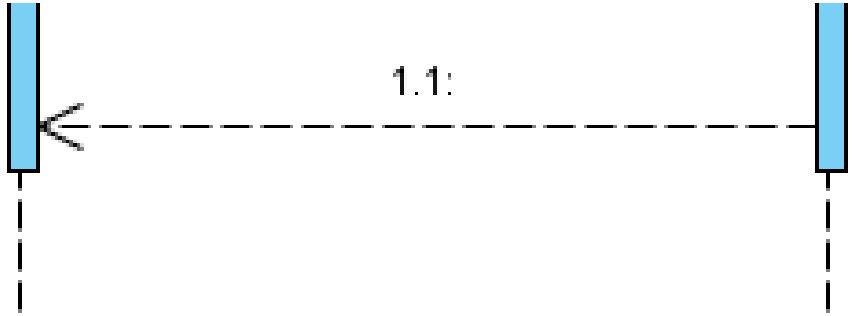
Messages: Indicates communication between lifelines of an interaction

Call message

- Represents invoking operation on the target lifeline
- Ie: Call a method on target lifeline

Return message

- Represents passing information back to the caller
- Ie: Return statement in method



https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/

Sequence diagram

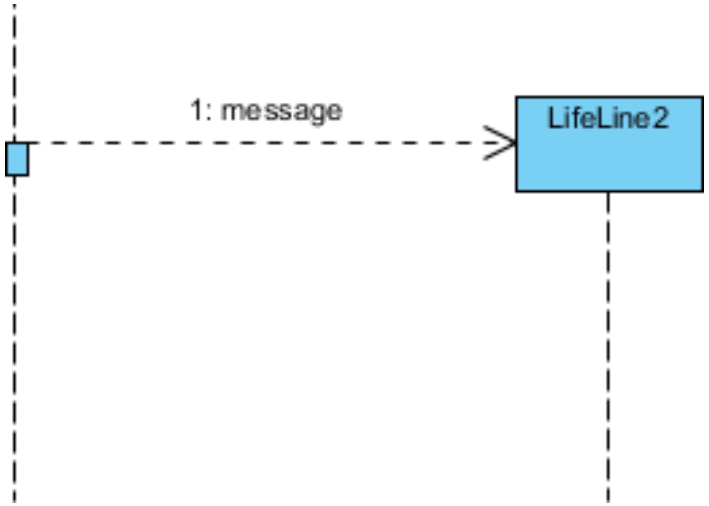
Messages: Indicates communication between lifelines of an interaction

Self message

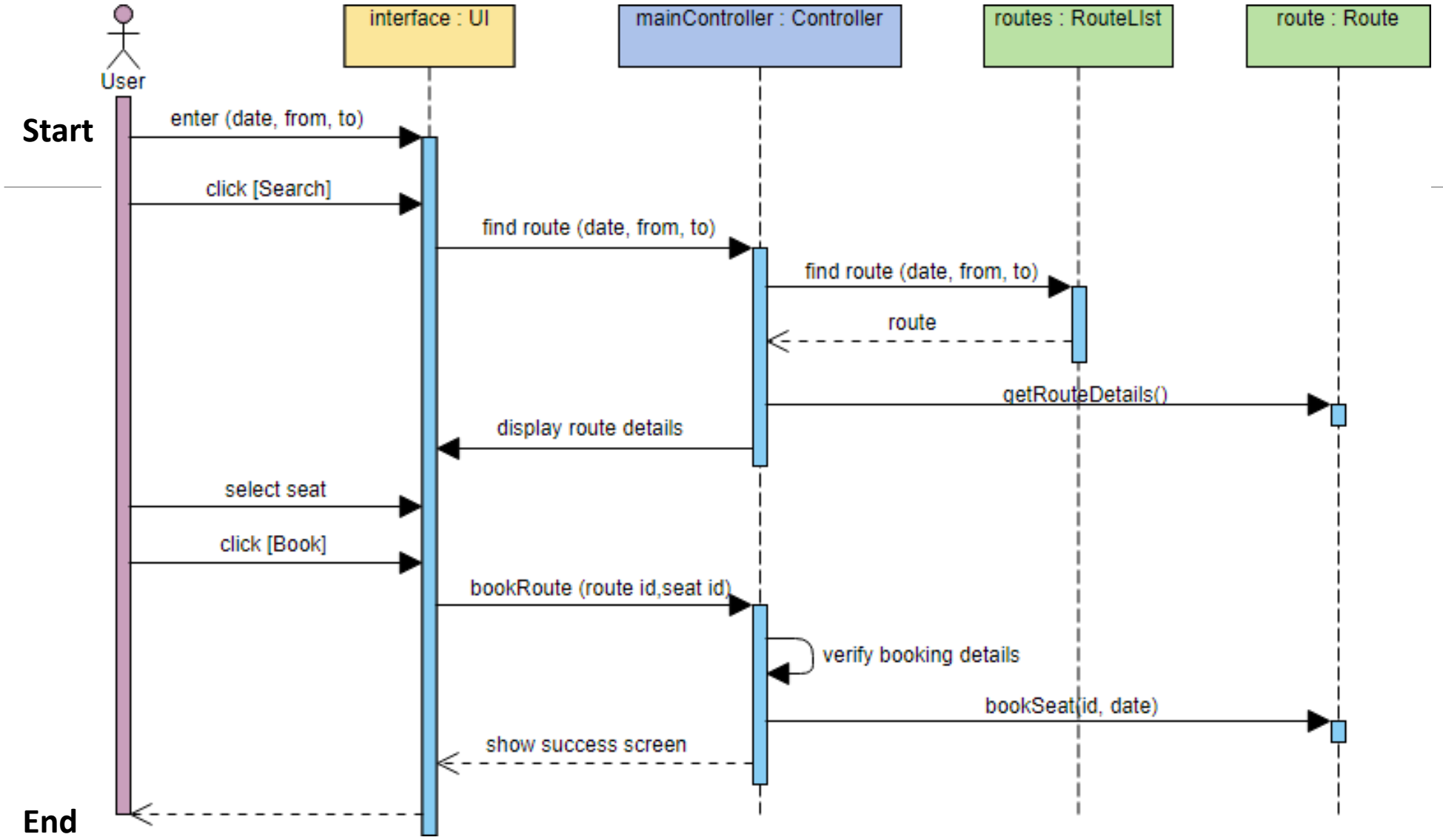
- Represents invoking message on the same lifeline

Create message

- Represents instantiation of the target lifeline



https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/


User
interface : UI
mainController : Controller
routes : RouteLIst
route : Route
Start
enter (date, from, to)
click [Search]
find route (date, from, to)
find route (date, from, to)
route
getRouteDetails()
display route details
select seat
click [Book]
bookRoute (route id,seat id)
verify booking details
bookSeat(id, date)
show success screen
End