DIS - Exercise 7

Artifact 5

2021/2022

Martin Radvanský

Artifact 2 - common problems

At least five actions in the main and alternative run.

What action mean?

Something what is done by your code.

- System validate input
- System display window with form.

What is not an action (code not exist):

- User fills textboxes
- User press button

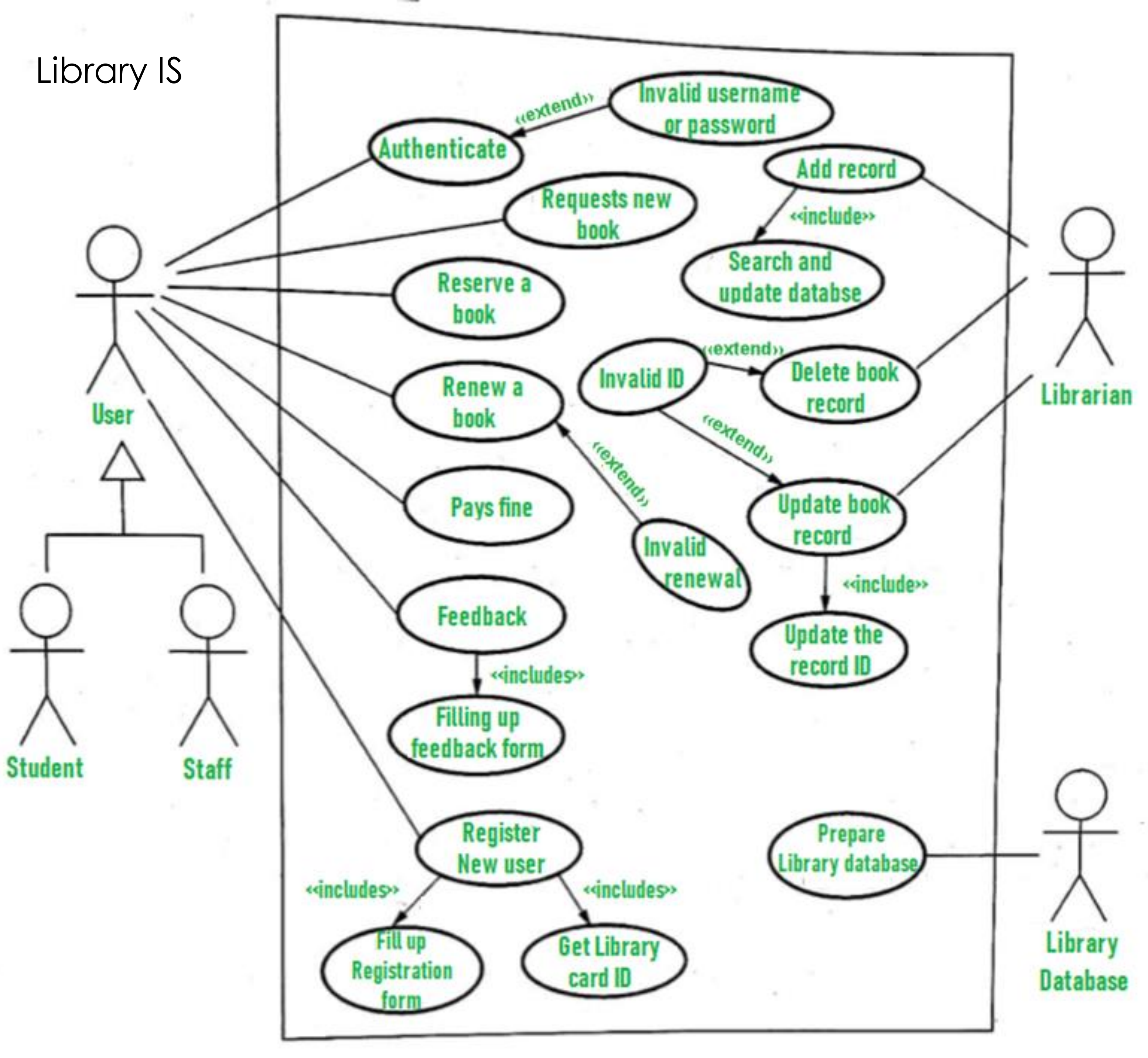
Actors and inheritance

Include/extend links

Active/passive actors

Activity diagrams decision box without decision text and Yes/No labels

Library IS


Authenticate
«extend»
Invalid username or password
Requests new book
Add record
«include»
Search and update databse
Reserve a book
«extend»
Invalid ID
Delete book record
Renew a book
«extend»
«extend»
Pays fine
Invalid renewal
Update book record
«include»
Update the record ID
Feedback
«includes»
Filling up feedback form
Register New user
«includes»
«includes»
Fill up Registration form
Get Library card ID
Prepare Library database
User
Student
Staff
Librarian
Library Database

Content of Artifact 3

- **One Chart of First Model of Domain** - Class diagram of main domain Classes, fields and methods, without data types – platform independent.
- **One Table of used resources** – Estimation of storage size, storage type.
- **½ page** of **Types of user interactions** with the system and estimation of their complexity.
- **½ page** First idea of the **system layout**.
- **½ page** Choice of the **used platform**.
- **½ page** Estimation of **patterns and architecture**.

Artifact 3 common problems

- Try to create rich first domain model.
  - Use names for links for better understanding relations
  - Use correct multiplicity labels 0,1,*
  - Thinks about attributes and methods
- Describe used platforms – OS, Software frameworks, web server, DB server, Internet browser. Write version from which your sw is supported. Like Windows 10 20H2 and above. .NET Framework 4.5, etc.
- Describe architecture 3-tier/layer, monolithical and why.
- Describe which patterns you expect to use. For domain model, for data layer.

Artifact 5 - **Domain model design**

- **Domain model -** extended conceptual model. Classes (methods, interface, data types)
- Relationships (completion)
- Interactions - static class diagram
- Sequence diagrams for the chosen functionality implementation.
- Design patterns (and architectural) used in the data, domain or presentation layer.

Artifact 5 - **Domain model design**

- Static class diagram (data + methods), association types, inheritance.

- Patterns used in domain model.

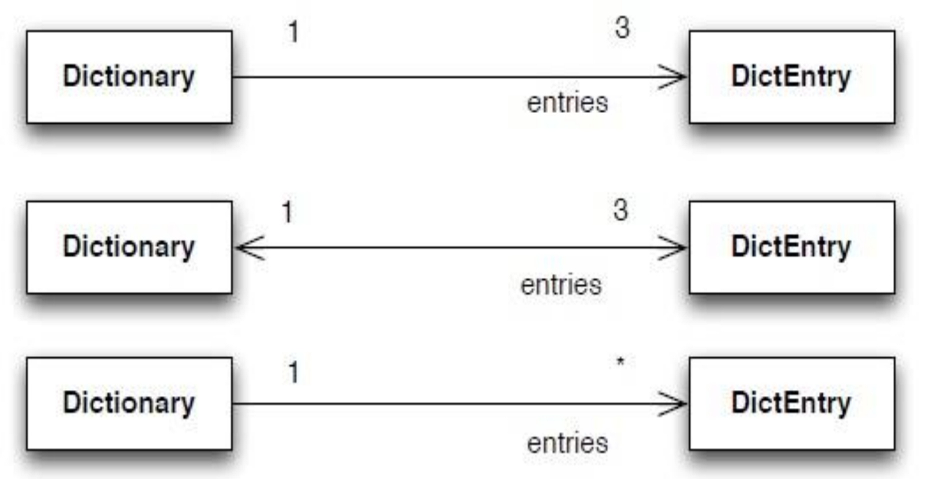
- It is possible to separate in diagrams the pure domain design from the design with patterns. Separate layers PL, BL, DAL for better readability.

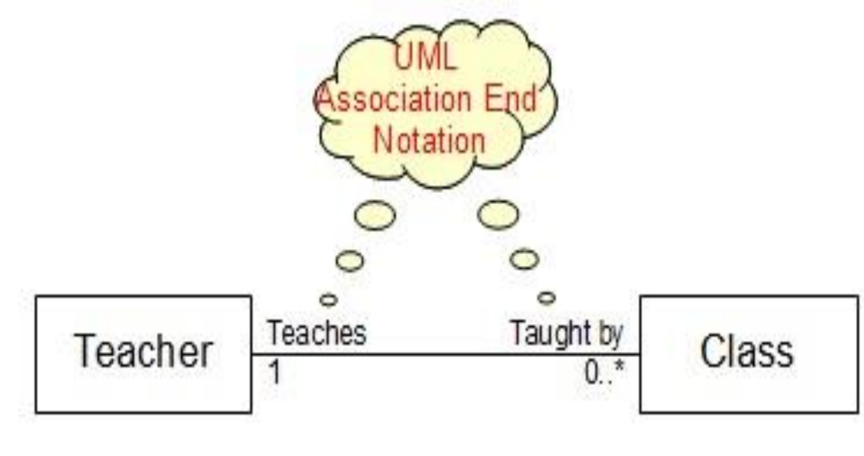
- Sequence diagrams for key operations (especially object cooperation)

Review of relations in the class diagram

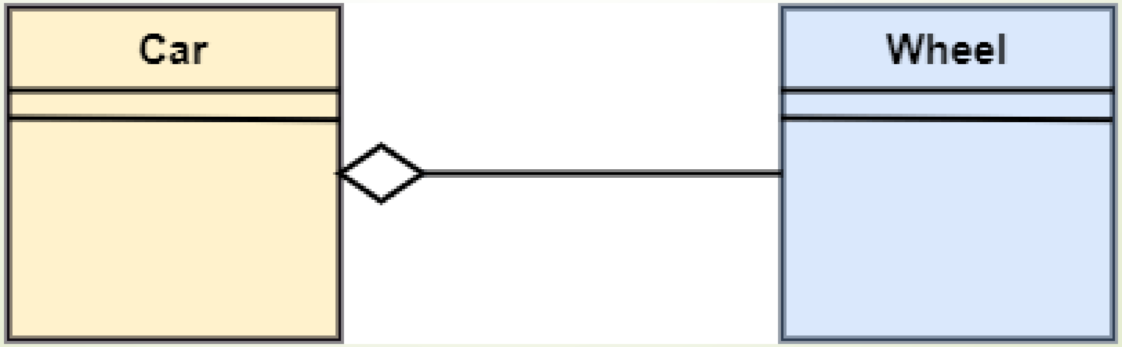
- The connections between objects cannot be created arbitrarily; they must be consistent with how the relationships between their classes are defined.

At the class level, connections correspond to **associations**; connections are actually instances of associations. Each association can be described in more detail by additional properties such as the name of the association, the role names of the associated classes, the multiplicity and the manageability of the association. The name of an association usually expresses, in a verb phrase, an action performed by one object using another object.

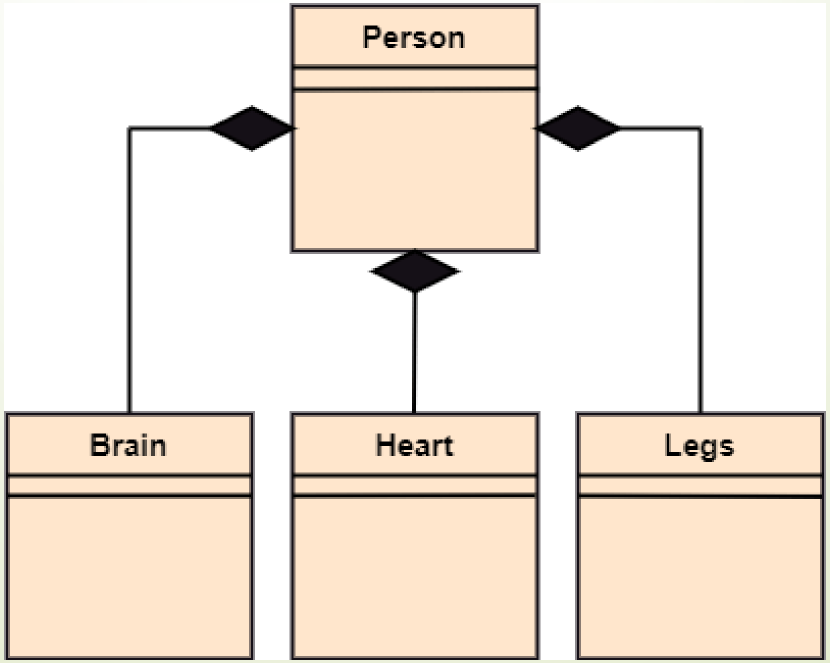


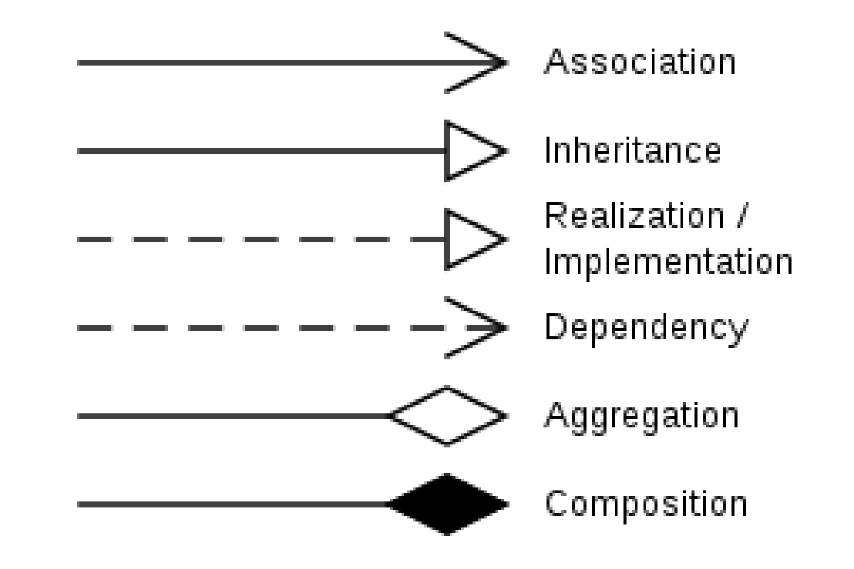


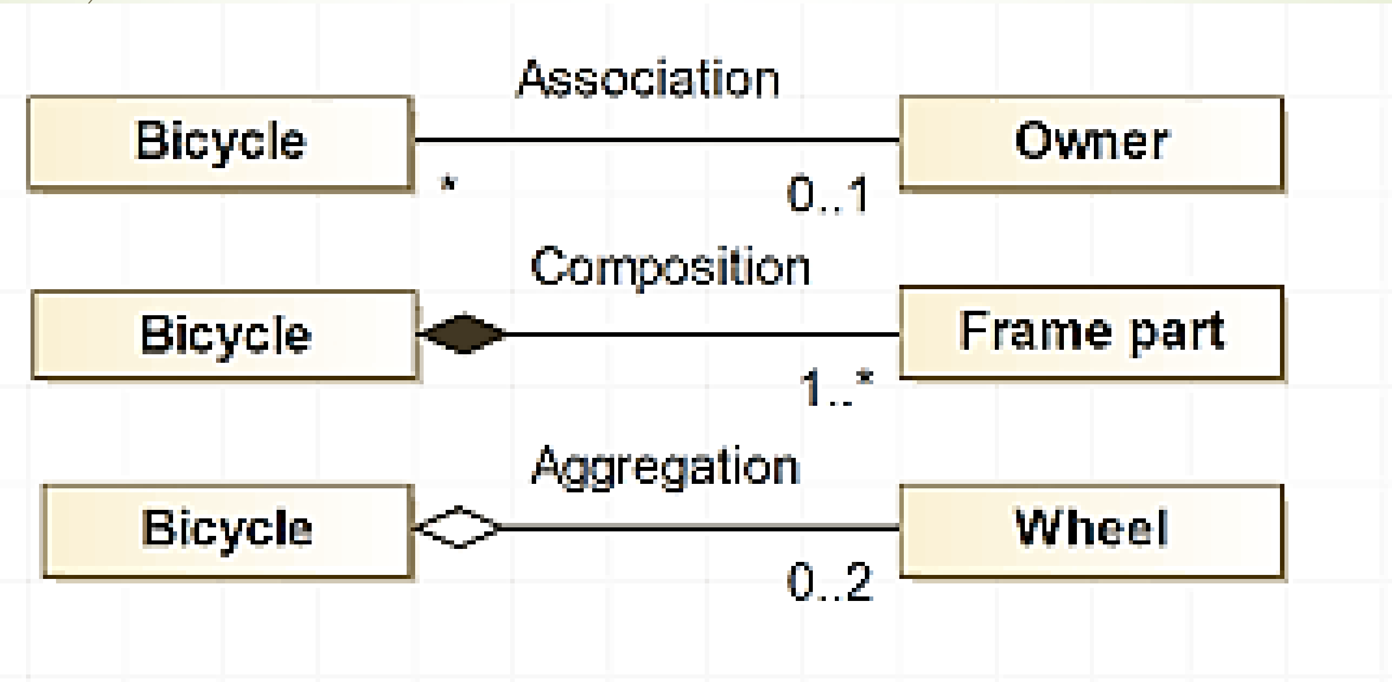
Aggregation represents a loose coupling between a whole and a part, where one object (the whole) uses the services of other objects (the parts). For example, the relationship between a car and wheels is an aggregation type relationship where the car and wheels form a single unit, but the wheel can exist even if it is not fitted to any car.



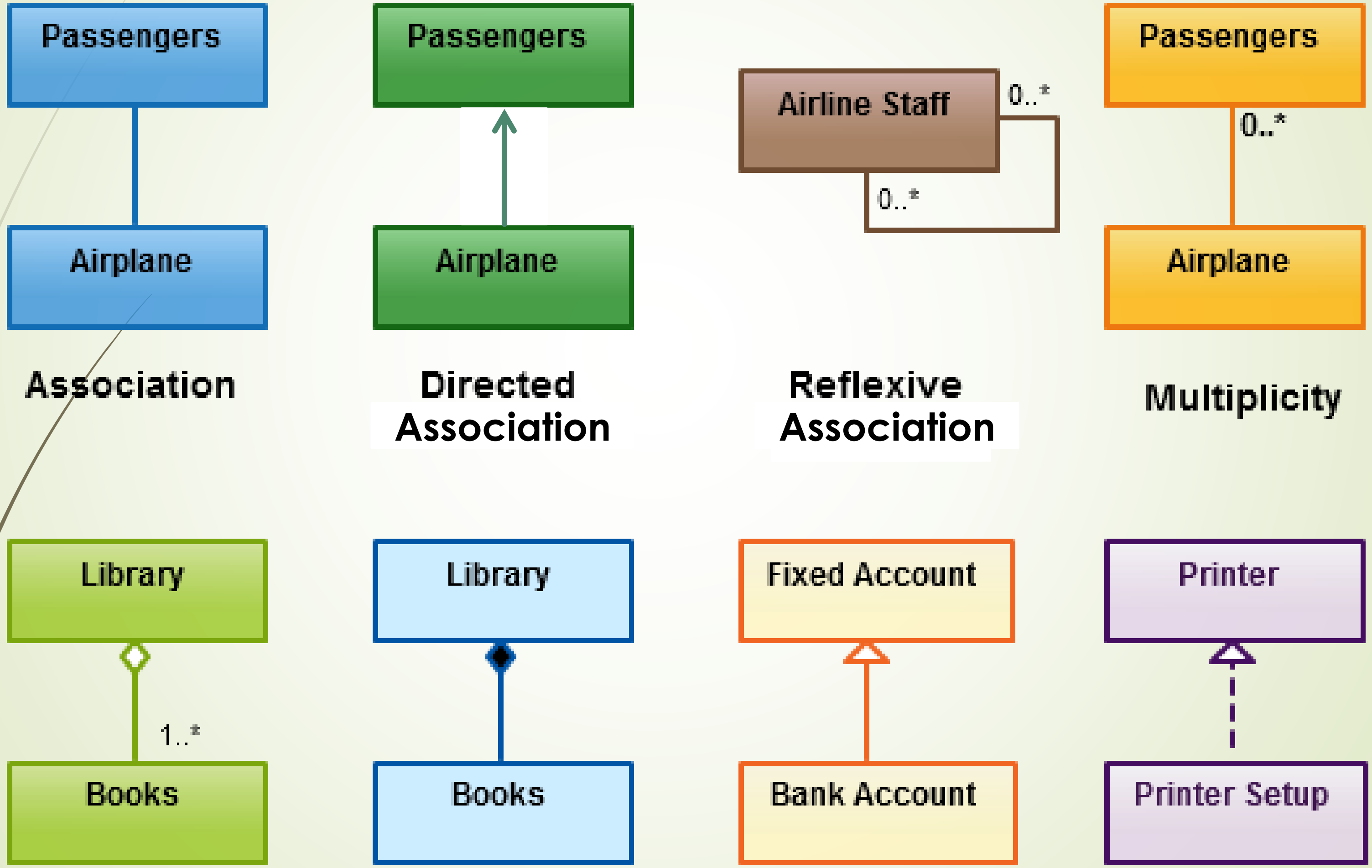
A stronger form of aggregation is **composition**. It is a relationship between a whole and a part, but this relationship is very close and does not allow the independent existence of a part without being attached to a whole. Moreover, unlike aggregation, the component must belong to a single whole and cannot be shared by multiple wholes.




Association
Inheritance
Realization / Implementation
Dependency
Aggregation
Composition


Association
Bicycle
Owner
*
0..1
Composition
Bicycle
Frame part
1..*
Aggregation
Bicycle
Wheel
0..2

UML class relations



Passengers

Airplane

Association

Passengers

Airplane

Directed Association

Airline Staff

0..*

0..*

Reflexive Association

Passengers

0..*

Airplane

Multiplicity

Library

1..*

Books

Aggregation

Library

Books

Composition

Fixed Account

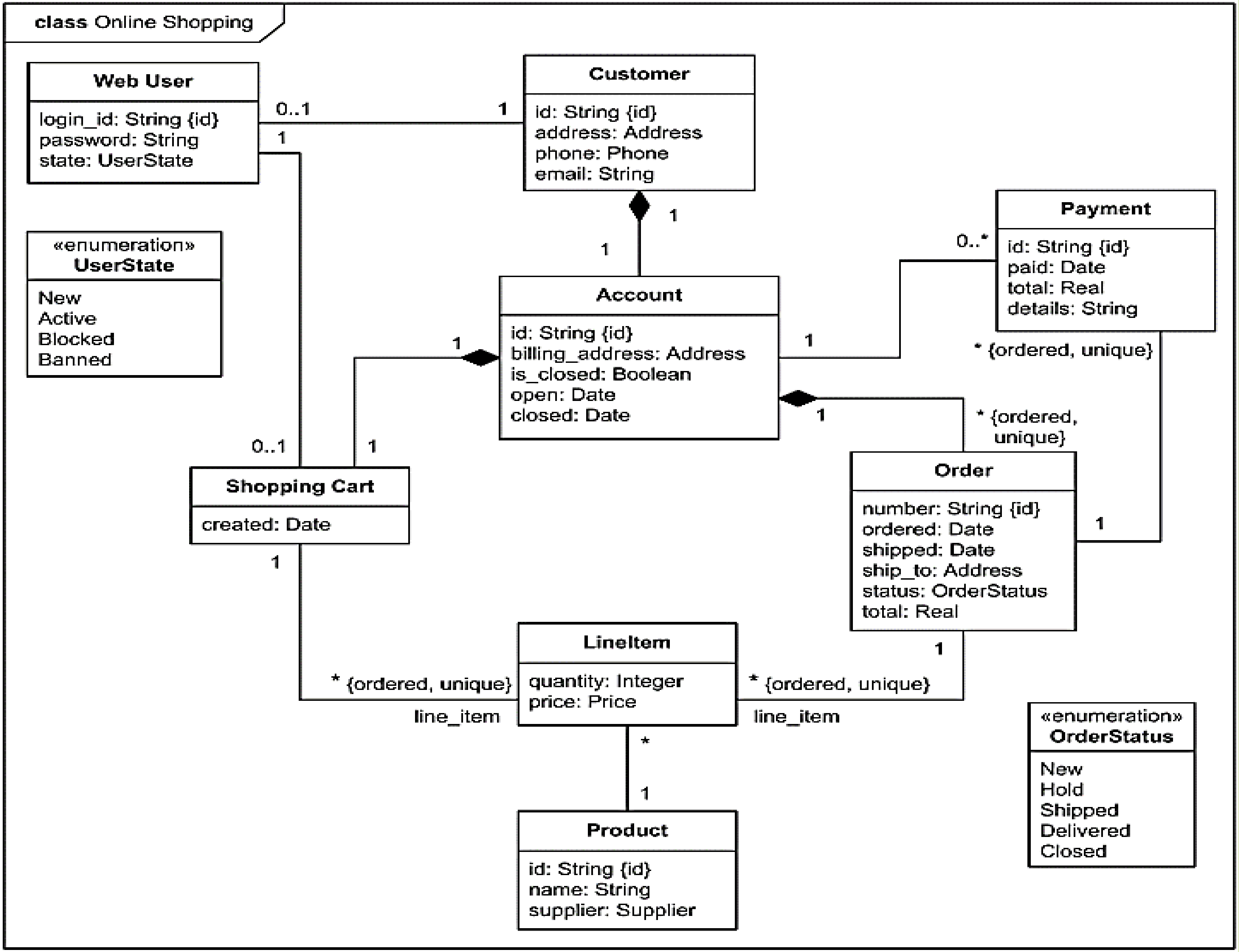
Bank Account

Inheritance

Printer

Printer Setup

Realization



class Online Shopping

Web User

login_id: String {id}
password: String
state: UserState

Customer

id: String {id}
address: Address
phone: Phone
email: String

«enumeration»
UserState

New
Active
Blocked
Banned

Payment

id: String {id}
paid: Date
total: Real
details: String

Account

id: String {id}
billing_address: Address
is_closed: Boolean
open: Date
closed: Date

Shopping Cart

created: Date

Order

number: String {id}
ordered: Date
shipped: Date
ship_to: Address
status: OrderStatus
total: Real

LineItem

quantity: Integer
price: Price

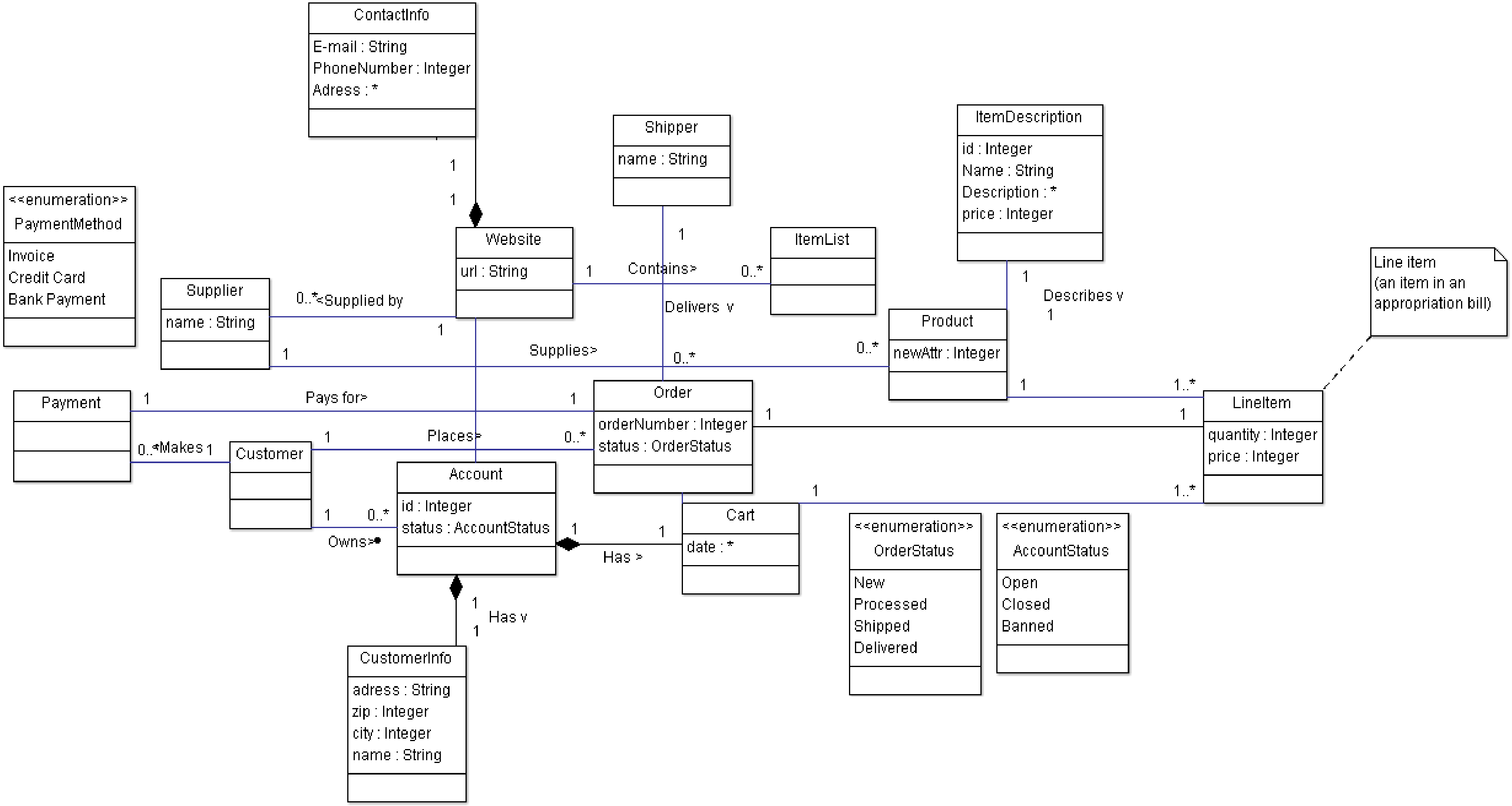
«enumeration»
OrderStatus

New
Hold
Shipped
Delivered
Closed

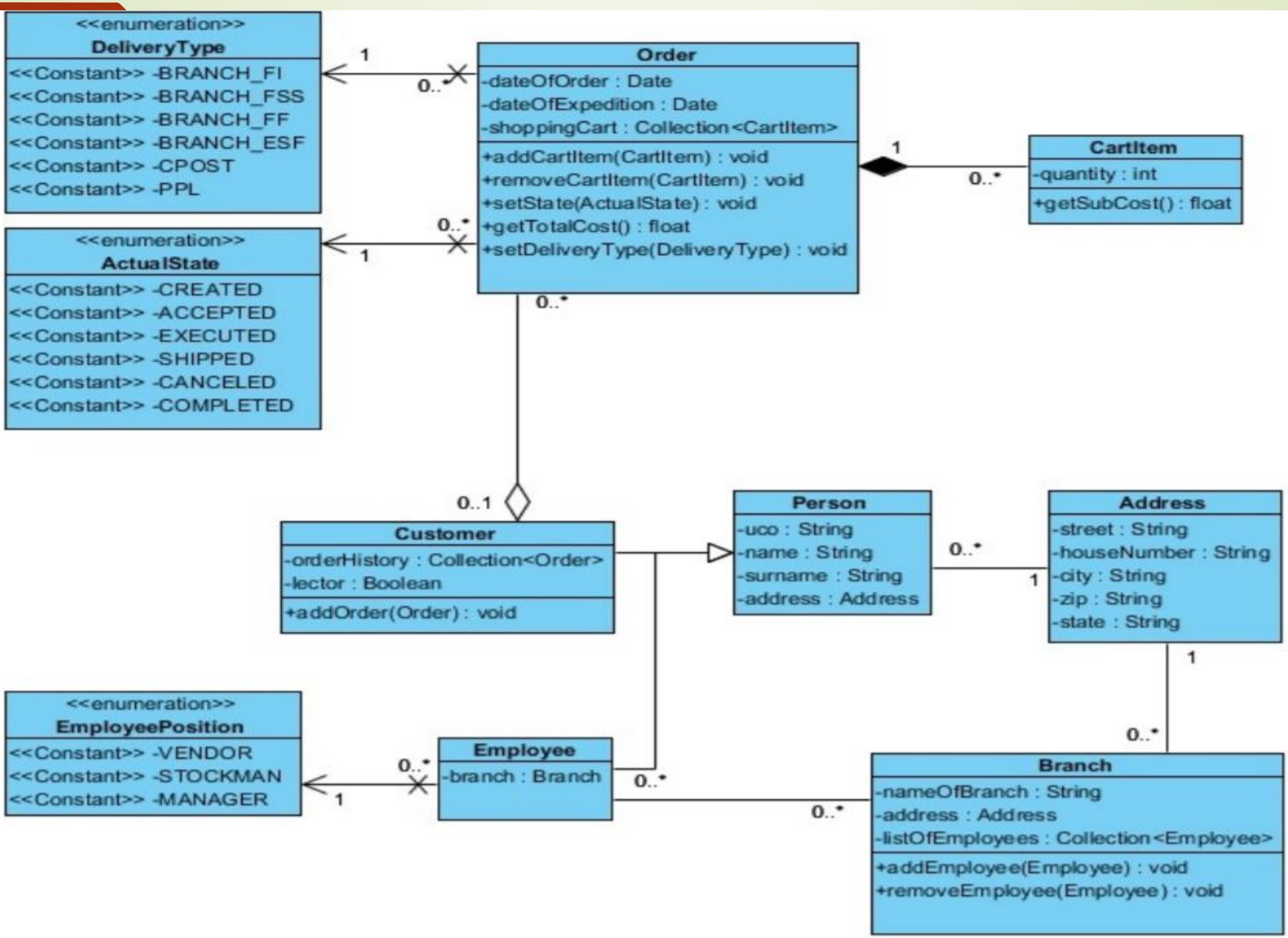
Product

id: String {id}
name: String
supplier: Supplier

Web shop


ContactInfo
E-mail : String
PhoneNumber : Integer
Adress : *
1
1
Shipper
name : String
ItemDescription
id : Integer
Name : String
Description : *
price : Integer
<<enumeration>>
PaymentMethod
Invoice
Credit Card
Bank Payment
Website
url : String
1
1
Contains>
0..*
ItemList
Line item
(an item in an
appropriation bill)
Supplier
name : String
0..*<Supplied by
1
Delivers v
1
Describes v
1
Product
newAttr : Integer
1
Supplies>
0..*
0..*
Payment
1
Pays for>
1
Order
orderNumber : Integer
status : OrderStatus
1
1..*
1
1
LineItem
quantity : Integer
price : Integer
0..*Makes
1
Customer
1
Places>
0..*
Account
id : Integer
status : AccountStatus
1
0..*
Owns>
1
Has >
1
Cart
date : *
1
1..*
<<enumeration>>
OrderStatus
New
Processed
Shipped
Delivered
<<enumeration>>
AccountStatus
Open
Closed
Banned
1
Has v
1
CustomerInfo
adress : String
zip : Integer
city : Integer
name : String

Domain model web order



<<enumeration>>
DeliveryType
<<Constant>> -BRANCH_FI
<<Constant>> -BRANCH_FSS
<<Constant>> -BRANCH_FF
<<Constant>> -BRANCH_ESF
<<Constant>> -CPOST
<<Constant>> -PPL

Order
-dateOfOrder : Date
-dateOfExpedition : Date
-shoppingCart : Collection<CartItem>
+addCartItem(CartItem) : void
+removeCartItem(CartItem) : void
+setState(ActualState) : void
+getTotalCost() : float
+setDeliveryType(DeliveryType) : void

CartItem
-quantity : int
+getSubCost() : float

<<enumeration>>
ActualState
<<Constant>> -CREATED
<<Constant>> -ACCEPTED
<<Constant>> -EXECUTED
<<Constant>> -SHIPPED
<<Constant>> -CANCELED
<<Constant>> -COMPLETED

Customer
-orderHistory : Collection<Order>
-lector : Boolean
+addOrder(Order) : void

Person
-uco : String
-name : String
-surname : String
-address : Address

Address
-street : String
-houseNumber : String
-city : String
-zip : String
-state : String

<<enumeration>>
EmployeePosition
<<Constant>> -VENDOR
<<Constant>> -STOCKMAN
<<Constant>> -MANAGER

Employee
-branch : Branch

Branch
-nameOfBranch : String
-address : Address
-listOfEmployees : Collection<Employee>
+addEmployee(Employee) : void
+removeEmployee(Employee) : void

Discussion